Spring 1994

Motion synthesis of mechanisms using constraint manifolds in image space

Yeou-Kai Wu
New Jersey Institute of Technology

Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

Order Number 9427002

Motion synthesis of mechanisms using constraint manifolds in image space

Wu, Yeou-Kai, Ph.D.

New Jersey Institute of Technology, 1994

U·M·I
300 N. Zeeb Rd.
Ann Arbor, MI 48106

ABSTRACT

MOTION SYNTHESIS OF MECHANISMS USING CONSTRAINT MANIFOLDS IN IMAGE SPACE

by
Yeou-Kai Wu

Kinematic mappings, quaternion algebra, and constraint manifolds in the algebraic image space are applied to the problems of the dimensional synthesis of mechanisms. Dimensions of a mechanism are determined such that a tracer frame fixed on the coupler will pass through or at least as close as possible to the desired positions and orientations in the physical space as the input link rotates about its fixed joint. First, using kinematic mappings, the desired positions and orientations of the tracer frame of the mechanism can be mapped onto points in a hyperspace in which the motion of the tracer frame can be represented by a curve. Second, using quaternion algebra, the structure equations representing the transformations from the reference frame to the tracer frame via each leg, each crank-coupler dyad of the mechanism, form the constraint manifolds of the mechanism. Finally, the problem of dimensional synthesis thus becomes one of finding a curve, generated by the intersection of constraint manifolds and fulfilling the constraint equations of kinematic mappings, which passes through or near the desired image points. The dimensions of the mechanism are found by using total least square algorithms to minimize the normal distance between all the desired image points and image curve of the tracer frame.

Using this approach, the synthesis problems of all three types of mechanisms, planar, spherical, and spatial, can be formulated similarly. It provides a straightforward tool for general motion synthesis problems. The theory is illustrated by numerical examples of planar and spherical mechanisms.

MOTION SYNTHESIS OF MECHANISMS USING CONSTRAINT MANIFOLDS IN IMAGE SPACE

by
Yeou-Kai Wu

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy

Department of Mechanical and Industrial Engineering

May 1994

APPROVAL PAGE

MOTION SYNTHESIS OF MECHANISMS USING CONSTRAINT MANIFOLDS IN IMAGE SPACE

Yeou-Kai Wu

Dr. Ian S. Fischer, Dissertation Advisor — Date
Associate Professor of Mechanical Engineering, NJIT

Dr. Rajesh N. Dave, Committee Member — Date
Associate Professor of Mechanical Engineering, NJIT

Dr. Anthony D. Rosato, Committee Member — Date
Associate Professor of Mechanical Engineering, NJIT

Dr. Avraham Harnoy, Committee Member — Date
Associate Professor of Mechanical Engineering, NJIT

Dr. Bruce G. Bukiet, Committee Member — Date
Assistant Professor of Mathematics, NJIT

BIOGRAPHICAL SKETCH

Author: Yeou-Kai Wu

Degree: Doctor of Philosophy Degree in Mechanical Engineering

Date: May 1994

Undergraduate and Graduate Education

- Doctor of Philosophy in Mechanical Engineering,
  New Jersey Institute of Technology,
  Newark, New Jersey, 1994

- Master of Science in Mechanical Engineering,
  Georgia Institute of Technology,
  Atlanta, Georgia, 1988

- Bachelor of Science in Mechanical Engineering,
  National Taiwan Institute of Technology,
  Taiwan, Republic of China, 1985

Major: Mechanical Engineering

Presentations and Publications:

Wu, Yeou-Kai, and Ian S. Fischer, "*Motion Synthesis of Mechanisms Using Constraint Manifolds,*" Accepted to Present at the 23rd ASME Design Technical Conference.

Wu, Yeou-Kai and Ian S. Fischer, "*Motion Synthesis of a Planar Mechanism Using Constraint Manifolds,*" Presented in Newark, New Jersey, August 20, 1993, at the 9th International Conference on CAD/CAM, Robotics, and Factories of the Future.

Wu, Yeou-Kai, "*Animating Rotation of an Object Using Quaternions,*" Presented in Polytechnic University, Brooklyn, New York, April 16, 1993, at the Third ASME Regional II Graduate Student Technical Conference.

Wu, Yeou-Kai, "*Synthesis of a Mechanism Using Quaternion Operators,*" Presented in Hoboken, New Jersey, April 16, 1992, at the Second ASME Region II Graduate Student Technical Conference.

This dissertation is dedicated to
my parents and
my lovely wife

ACKNOWLEDGMENT

The author will always remain indebted to his advisor, Professor Ian Fischer, for his ever present guidance and support, both technical and interpersonal, throughout this research.

Thanks are due to Professors Avraham Harnoy, Rajesh Dave, Anthony Rosato, and Bruce Bukiet for dedicating their time to serve as members of the committee.

The author is grateful to Professors Ronald Kane, Harry Herman, and Ronyaw Chen for graduate advice and assistance throughout these years.

The author also appreciates the technical staff in the Department of Mechanical Engineering: Don Rosander, Joe Glaz, and Dave Singh for their time-to-time assistance.

Sincere thanks are due to Professor Janet Bodner for patiently correcting the English pronunciation and grammar. Her friendship and hospitality to international students are greatly appreciated.

Professor Lester Ingber, Professor Nicol Schraudolph, and Dr. Mohan Bodduluri unselfishly provided their computer software codes.

Timely help and suggestions came from many good friends: Dr. Herli Surjanhata, Dr. Suat Ozsoylu, Philip Remington, Sahidur Rahman, Shu-Chieh(Roger) Chang, Hsin-Te Liao, Jerry Volcy, and Hermean Wong.

The author is grateful to the members of his family for their love, faith, and patience.

Lastly, the author would like to express his greatest and deepest gratitude to his lovely wife, Li-hsiang(Lila) Wu, for the many sacrifices she has made for his education that can not be compensated in any way.

TABLE OF CONTENTS

Chapter **Page**

Chapter **Page**

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

CHAPTER 1

INTRODUCTION

1.1 Summary of Research

A general method for motion synthesis of a mechanism has been developed. The method applies the theories of kinematic mappings, quaternion algebra, and constraint manifolds to find the dimensions of a mechanism which will guide a tracer frame on the coupler to go through or as close as possible to the desired positions, depending on the number of positions, in the physical space. The dimensions to be found include the locations of the fixed pivots, the link lengths of the input link and the output link, the dimensions of the coupler, and the orientation of the tracer frame on the coupler with respect to the reference frame.

Three types of mechanisms, planar, spherical, and spatial, are investigated in this study. Numerical examples are presented to illustrate the method. For planar mechanisms, both numerical and graphical results are presented. For spherical mechanisms, numerical results are shown. A conceptual discussion of this approach applied to generalized spatial mechanisms is also included in this research.

Many theories have been developed to solve motion synthesis problems. Some are for special mechanisms, while others are for generalized ones. They all have their own assets and liabilities. The method presented in this study involves a search for the minimization of a nonlinear least square function, which requires an initial guess for the set of the design parameters. Methods for generating proper initial guesses play an important role in this numerical approach. For this reason, three algorithms have been tested. One is the well known Levenberg-Marquart algorithm, and the others are newly developed artificial intelligence searching algorithms: genetic algorithms and simulated annealing methods.

1.2 Objective

The purpose of this dissertation is to combine the motion synthesis problem with the theory of constraint-manifold equations developed by McCarthy (1990) while using the concept of kinematic mappings proposed by Ravani and Roth (1982) to provide a straightforward and general approach to solve motion synthesis problems.

1.3 Background

Dimensional synthesis of a mechanism is the process of designing a mechanism to perform a desired task. Dimensional synthesis of mechanisms can be classified into three different categories (Sandor and Erdman, 1988):

1. Function generation: the motion of a body (the output link) is to change in a specified manner with respect to the motion of another body (the input link). Usually the motion of the output link can be expressed as a specified function of the motion of the input link.
2. Motion generation, or rigid-body guidance: a body (the floating link) is to pass through or as close as possible to prescribed positions and orientations in physical space.
3. Path generation: a point on a body is to trace a specified path in physical space.

This dissertation will focus only on the motion synthesis of mechanisms (category 2). Motion synthesis is used to find dimensions of a mechanism which guides a body, or a tracer frame on the floating link, through prescribed positions and orientations in physical space. There are schemes for both exact-position synthesis and approximate-position synthesis.

Exact-position synthesis of planar motion was first developed graphically by Burmester (1888). Using complex number theory, Freudenstein and Sandor (1961) developed an algebraic method for studying this procedure analytically. This work has further been developed by Bottema and Roth (1979), Sandor and Erdman (1988), and

Mirth (1992) among many others. Both graphical and analytic methods can synthesize up to five prescribed positions for the motion synthesis problem. By adjusting some of the design parameters, for example crank lengths, exact-position motion synthesis of more than five positions is possible. Two-phase motion synthesis has been developed by Wilhelm (1989) by adjusting fixed pivots or by adjusting crank lengths for up to five prescribed positions. Using the same approach, three-phase motion synthesis has been investigated by Wang (1993) for up to seven prescribed positions. This approach, however, increases the degree of freedom by one, therefore, requiring one more input. Exact-position synthesis of spherical motion generations has been studied by Suh and Radcliffe (1978), Dowler, Duffy, and Tesar (1976), and many others. Computer software packages, Lincages-4© (1988) and *Sphinx* (Lorachelle et al, 1993), are now available for synthesis of planar four-bar and spherical four-bar mechanisms, but both packages are limited to synthesis problems with only up to four prescribed positions. Synthesis of spatial motion generations can be found in Sandor, Xu, and Yang (1986), Sandor, Weng, and Xu (1988), Youm and Huang (1990), and Subbian and Flugrad (1992) et al. The synthesis of special mechanisms has been studied by Jenuwide and Midha (1992) and others.

If the prescribed situations need only be approximated, then the design of a mechanism is much facilitated. Planar approximate motion synthesis has been developed by Sarkisyan, Gupta, and Roth (1973), Gupta and Roth (1975). Mixed exact-approximate position synthesis of a planar mechanism in which the trace frame will pass certain important precision positions and approximate the rest can be found in Sutherland (1977). Selective precision synthesis methods used to achieve the prescribed limits of precision, called the *accuracy neighborhoods*, have been developed by Kramer and Sandor (1975) and Premkumar, Dhall, and Kramer (1988) and others. Using kinematic mappings, Ravani and Roth (1983) first introduced the idea of curve fitting in the image space for the dimensional synthesis of planar motions. They used the intersection of two general

geometric equations to represent a general motion in physical space which can be mapped into a curve in image space for the synthesis of four-bar mechanisms. Bodduluri and McCarthy (1993) used a similar approach for synthesis of spherical four-bar mechanisms.

After all these previous works, there is no general approach available that would work with three-dimensional mechanisms. The proposed method based on the coordinate transformations to form the constraint manifolds provides a general approach for the motion synthesis problems.

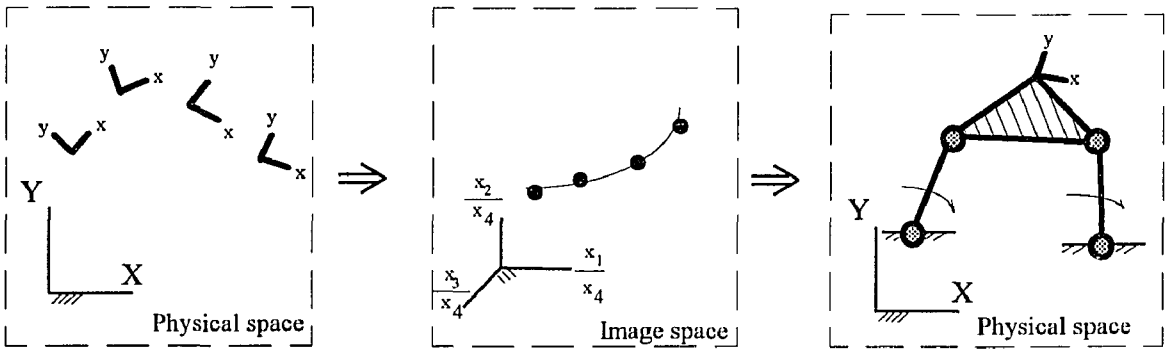


Figure 1-1 The kinematic mapping between physical and image spaces.

In the problem of motion synthesis, we prescribe several situations, several combinations of displacements and orientations relative to a reference frame, for the tracer frame of the mechanism. Dimensions of a mechanism are determined such that a tracer frame fixed on the coupler will pass through or at least as close as possible to the desired situations in physical space as the input link rotates about its fixed joint. First, using kinematic mappings, the desired positions and orientations of the tracer frame of the mechanism can be mapped onto points in a hyperspace, called the *image space* (Ravani and Roth, 1983), in which the motion of the tracer frame can be represented by a curve. Second, using quaternion algebra, the structure equations representing the coordinate transformations from the reference frame to the tracer frame via each leg, each crank-coupler dyad of the mechanism, form the constraint manifolds of the mechanism. Finally,

the problem of dimensional synthesis becomes one of finding a curve generated by the intersection of constraint manifolds, which passes through or near the desired image points while fulfilling the constraint equations of kinematic mappings. The dimensions of the mechanism are found by optimization algorithms to minimize the normal distance between all the desired image points and the image curve of the tracer frame. Details of curve fittings in the image space have been studied by Ravani and Roth (1982). The idea is illustrated in Figure 1-1.

Using this approach, the synthesis problems of all three types of mechanisms, planar, spherical and spatial, can be formulated similarly. It provides a straightforward method for the general motion synthesis problems.

1.4 Outline of Dissertation

In chapter 2, three types of displacements, planar, spherical and spatial, and their corresponding kinematic mappings in quaternion forms, are discussed. In chapter 3, quaternion algebra is used to construct the structure equations of open chain mechanisms suitable for the motion synthesis problems. In chapter 4, we discuss the normal curve fitting techniques in the image space to formulate the error functions to be minimized using global minimization methods. In chapter 5, global minimization methods are discussed in more detail. In chapter 6, we present some examples and results using different global minimization methods. In chapter 7, conclusions and discussions are presented.

CHAPTER 2

DISPLACEMENTS AND QUATERNIONS

2.1 Introduction

Displacements can be classified into three categories: planar, spherical, and spatial. Planar displacements are motions limited to a plane, or a two-dimensional space. Spherical displacements, or rotations, are motions rotated about a fixed point. Spatial displacements, the most general type of motions, are neither limited to a plane nor rotated about a fixed point. They can be expressed as screw motions about an axis in three-dimensional space. For each category, we can construct its corresponding type of mechanism for the motion synthesis problem. The use of matrices to represent displacements, or rigid transformations, is common and well known, especially in robotics, but the focus of this study is on the usage of various quaternions to describe these displacements in a higher-dimension space, called the image space (Ravani and Roth, 1983). In this chapter, we present the relationships between these displacements and various quaternions. For a planar displacement, there is always a point that does not move during the displacement. This point is called the pole of the planar displacement. For a spherical displacement, there is an axis of rotation that does not change its position. Similarly, for a spatial displacement, we have a screw axis that maintains its position during a spatial displacement. It is the pole of a planar displacement, the axis of rotation of a spherical displacement, and the screw axis of a spatial displacement that can be arranged into special hypercomplex numbers, called *planar quaternions*, *quaternions*, and *dual quaternions*, respectively, which can be used instead of matrices to represent coordinates transformations (McCarthy 1990).

McCarthy has also shown that these various quaternions can be constructed by using the theory of Clifford algebra. Planar and dual quaternions are proved to be Clifford

algebras for three and four dimensional spaces, but with degenerate scalar products, while quaternions are also elements of the Clifford algebra of three-dimensional space with the usual Euclidean scalar products. The desired positions and orientations of the tracer frame, when written in quaternion forms, can be mapped onto points in the image space. The coordinate transformations from the reference frame to the tracer frame can be obtained by the multiplication of these various quaternions to represent a structure equation. Therefore, these various quaternions play two roles in this method:

- to map the prescribed situations onto points in the image algebraic spaces;
- to construct the structure equations from the reference frame to the tracer frame.

The quaternion, a four-element entity, was introduced by William R. Hamilton in 1843, who defined a quaternion in the following way:

$$q = w + \vec{i}\,x + \vec{j}\,y + \vec{k}\,z \tag{2-1}$$

or in terms of one scalar w and one vector v

$$q = (w, v) \tag{2-2}$$

where $\vec{i}$, $\vec{j}$, and $\vec{k}$ are the three imaginary units. This definition looks quite similar to the definition of a vector as a complex number

$$v \quad x + \vec{i}\,y \tag{2-3}$$

in the complex plane. A quaternion can be seen as the complex representation of a point (x, y, z, w) in some 4-dimensional space, which is governed by the following fundamental rules:

$$\begin{aligned} \vec{i}^{\,2} = \vec{j}^{\,2} = \vec{k}^{\,2} = \vec{i}\,\vec{j}\,\vec{k} &= -1, \\ \vec{j}\,\vec{k} &= -\vec{k}\,\vec{j} = \vec{i}, \\ \vec{k}\,\vec{i} &= -\vec{i}\,\vec{k} = \vec{j}, \\ \vec{i}\,\vec{j} &= -\vec{j}\,\vec{i} = \vec{k}. \end{aligned} \tag{2-4}$$

2.2 Planar Displacements and Planar Quaternions

2.2.1 Planar Displacement of a Rigid Body

The planar displacement of a rigid body can be represented by a translation $\boldsymbol{d} = [a, b]^T$ of coordinate frame M fixed on the moving body and a rotation θ of the moving body in a plane. The coordinate transformation equation which relates a point p in the moving coordinate frame M to those in a fixed reference coordinate frame F (Figure. 2-1) is:

$$X = [A]\,x + \boldsymbol{d} \tag{2-5}$$

or

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} a \\ b \end{bmatrix} \tag{2-6}$$

where

$$[A] = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \quad \boldsymbol{d} = \begin{bmatrix} a \\ b \end{bmatrix}$$

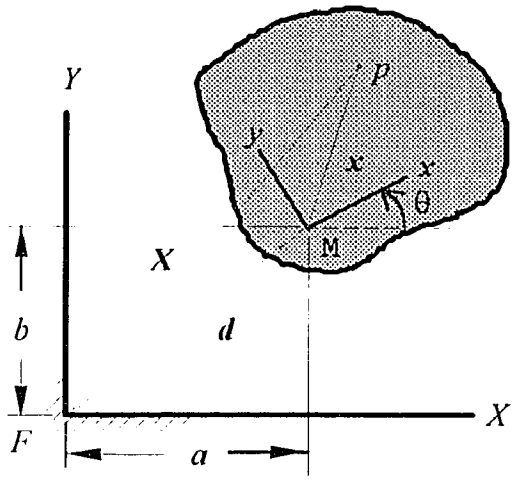


Figure 2-1 A planar displacement of a rigid body.

or in terms of homogeneous coordinates

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & a \\ \sin\theta & \cos\theta & b \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{2-7}$$

The coordinate vectors $\boldsymbol{X} = [X\ Y]^T$ and $\boldsymbol{x} = [x, y]^T$ represent points in the planar reference coordinate frames F and M, respectively. The parameters a, b, and θ represent a planar displacement of the moving frame M relative to the reference fixed frame F.

The components of the pole of a planar displacement are coefficients required to formulate the planar quaternion which can be used instead of transformation matrix of eq.(2-5) to represent a displacement. We also discuss the pole of a planar displacement. For a general planar displacement, there is always one point in the plane that does not change its position during the displacement. This means that the position coordinates of this particular point remain the same. The moving frame can be considered as having rotated about this point in the fixed frame. Therefore, its position coordinates are the same in both fixed frame F and moving frame M. This point is called the *pole* of the planar displacement (Suh, 1978). The pole of the planar displacement is actually one of the eigenvectors for the system of eq.(2-7). The coordinate of the pole $\boldsymbol{P}$ can be obtained by solving eq.(2-5):

$$\boldsymbol{P} = [A]\ \boldsymbol{P} + \boldsymbol{d} \tag{2-8}$$

or

$$\begin{bmatrix} p_x \\ p_y \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} p_x \\ p_y \end{bmatrix} + \begin{bmatrix} a \\ b \end{bmatrix}$$

Solving for $\boldsymbol{P}$ we have

$$\boldsymbol{P} = [\boldsymbol{I}\text{-}\boldsymbol{A}]^{-1}\boldsymbol{d} \tag{2-9}$$

or

$$p_x = (\frac{a}{2}\sin\frac{\theta}{2} - \frac{b}{2}\cos\frac{\theta}{2})/\sin\frac{\theta}{2},$$
$$p_y = (\frac{a}{2}\cos\frac{\theta}{2} + \frac{b}{2}\sin\frac{\theta}{2})/\sin\frac{\theta}{2}. \quad (2\text{-}10)$$

2.2.2 Planar Quaternions

In the last section, planar displacements were described by using matrix transformations. A planar displacement can also be expressed by a planar quaternion which is an even Clifford algebra of three-dimensional space with special degenerate scalar product. A typical form of a planar quaternion can be written as a four-element entity, $Q = q_4 + \varepsilon q_1 \vec{i} + \varepsilon q_2 \vec{j} + q_3 \vec{k}$, where ε is the dual number, with the properties of $\varepsilon \neq 0$, but $\varepsilon^2 = 0$ (Porteous, 1981). A general planar displacement can be characterized by the quaternion operator:

$$Q(a, b, \theta) = \cos\frac{\theta}{2} + \varepsilon\frac{1}{2}(a\cos\frac{\theta}{2} + b\sin\frac{\theta}{2})\vec{i} + \varepsilon\frac{1}{2}(b\cos\frac{\theta}{2} - a\sin\frac{\theta}{2})\vec{j} + \sin\frac{\theta}{2}\vec{k} \quad (2\text{-}11)$$

or simplified as

$$= \cos\frac{\theta}{2} + \varepsilon p_y \sin\frac{\theta}{2}\vec{i} - \varepsilon p_x \sin\frac{\theta}{2}\vec{j} + \sin\frac{\theta}{2}\vec{k} \quad (2\text{-}12)$$

where lengths (a, b) denotes the distances through which the moving frame translates in the original directions of the unit vectors $\vec{i}$ and $\vec{j}$, and angle θ indicates the rotation of the moving frame about the $\vec{k}$ axis, and point (p_x, p_y) is the pole of the displacement. The components of quaternion Q are:

$$q_1 = p_y \sin\frac{\theta}{2} = \frac{1}{2}(a\cos\frac{\theta}{2} + b\sin\frac{\theta}{2}),$$
$$q_2 = -p_x \sin\frac{\theta}{2} = \frac{1}{2}(b\cos\frac{\theta}{2} - a\sin\frac{\theta}{2}),$$
$$q_3 = \sin\frac{\theta}{2},$$
$$q_4 = \cos\frac{\theta}{2}. \quad (2\text{-}13)$$

We now define the mappings for planar kinematics, which can be used to map the desired positions and orientations of the tracer frame into points in the image space for the synthesis problems. The mapping is

$$q_1 : q_2 : q_3 : q_4 = \frac{1}{2}\left(a\cos\frac{\theta}{2} + b\sin\frac{\theta}{2} \right) : \frac{1}{2}\left(b\cos\frac{\theta}{2} - a\sin\frac{\theta}{2} \right) : \sin\frac{\theta}{2} : \cos\frac{\theta}{2} \qquad (2\text{-}14)$$

Positions with $\theta = 0$ will map into points at $\left(\frac{a}{2}, \frac{b}{2}, 0, 1 \right)$, while positions with $\theta = \pi$ will correspond to image points at $\left(\frac{b}{2}, -\frac{a}{2}, 1, 0 \right)$. The reference frame, at $(a = b = \theta = 0)$, maps into the point (0, 0, 0, 1) of the planar quaternion image space. In general, the points in the image space of planar displacements satisfy the algebraic relation:

$$q_3^2 + q_4^2 = 1 \qquad (2\text{-}15)$$

Other forms of planar kinematic mappings are described in Blaschke and Müller (1956) and Ravani (1982).

The planar displacement quantities (a, b, θ) can be recovered from q_i, i = 1,2,..4 by the following formula:

$$\begin{aligned} a &= 2\,(q_1q_4 - q_2q_3), \quad & \sin\theta &= 2\,q_3q_4, \\ b &= 2\,(q_1q_3 + q_2q_4), \quad & \cos\theta &= q_4^2 - q_3^2. \end{aligned} \qquad (2\text{-}16)$$

Eq.(2-7), in terms of q_i, becomes

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} q_4^2 - q_3^2 & -2q_3q_4 & 2(q_1q_4 - q_2q_3) \\ 2q_3q_4 & q_4^2 - q_3^2 & 2(q_1q_3 - q_2q_4) \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad (2\text{-}17)$$

2.2.3 Planar Quaternion Product

The structure equation representing the transformation from the reference frame to the tracer frame is obtained by successive coordinate frame transformations. The successive coordinate frame transformations can be expressed in the form of the successive products of planar quaternions. The planar quaternion $Q = q_4 + \varepsilon q_1 \vec{i} + \varepsilon q_2 \vec{j} + q_3 \vec{k}$ can also be

written as a four dimensional vector $Q = (q_1, q_2, q_3, q_4)$. The product of two planar quaternions $Q = q_4 + \varepsilon q_1 \vec{i} + \varepsilon q_2 \vec{j} + q_3 \vec{k}$ and $H = h_4 + \varepsilon h_1 \vec{i} + \varepsilon h_2 \vec{j} + h_3 \vec{k}$ results in a product of their associated vectors, QH.

$$\begin{aligned} QH &= (q_4 + \varepsilon q_1 \vec{i} + \varepsilon q_2 \vec{j} + q_3 \vec{k}).(h_4 + \varepsilon h_1 \vec{i} + \varepsilon h_2 \vec{j} + h_3 \vec{k}) \\ &= - q_3 h_3 + q_4 h_4 \\ &\quad + \varepsilon (q_4 h_1 - q_3 h_2 + q_2 h_3 + q_1 h_4) \vec{i} \\ &\quad + \varepsilon (q_3 h_1 + q_4 h_2 - q_1 h_3 + q_2 h_4) \vec{j} \\ &\quad + (q_4 h_3 + q_3 h_4) \vec{k} \end{aligned} \tag{2-18}$$

The components of QH can be obtained by representing the Clifford algebra in matrix form (McCarthy 1990). From eq(2-18): $QH = [Q^+]H$,

$$[Q^+]H = \begin{bmatrix} q_4 & -q_3 & q_2 & q_1 \\ q_3 & q_4 & -q_1 & q_2 \\ 0 & 0 & q_4 & q_3 \\ 0 & 0 & -q_3 & q_4 \end{bmatrix} \begin{Bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \end{Bmatrix} \tag{2-19}$$

or

$QH = [H^-]Q$

$$[H^-]Q = \begin{bmatrix} h_4 & h_3 & -h_2 & h_1 \\ -h_3 & h_4 & h_1 & h_2 \\ 0 & 0 & h_4 & h_3 \\ 0 & 0 & -h_3 & h_4 \end{bmatrix} \begin{Bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{Bmatrix} \tag{2-20}$$

2.3 Spherical Displacements and Quaternions

We next present the spherical displacements. Ravani (1982) used the Euler parameters of an orthogonal rotation transformation matrix to define a mapping of the matrix into a three-dimensional projective space to study spherical kinematics. Spherical displacements can be identified with elements of the Clifford algebra of three-dimensional space with the

usual Euclidean scalar products by assembling the Euler parameters of a rotation into the quaternion.

2.3.1 Spherical Displacement of a Rigid Body

The rotation of a rigid body about a fixed point, termed a spherical displacement, can be represented by a matrix transformation between three dimensional reference frames M and F (Figure. 2-2).

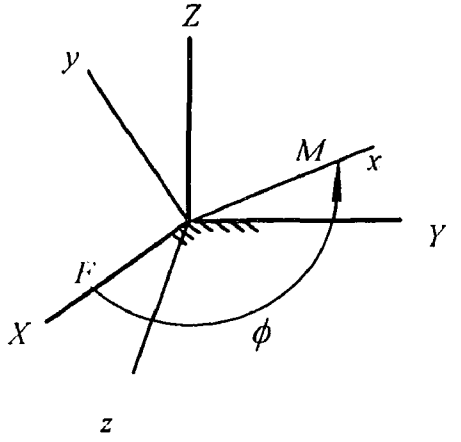


Figure 2-2 A spherical displacement of a rigid body.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{2-21}$$

The vector $X=[X, Y, Z]^T$ and $x=[x, y, z]^T$ represent points in frames F and M, respectively. The rotation matrix $[A] = a_{ij}$, $i,j = 1, 2, 3$, is orthogonal and depends on three parameters, namely three Euler angles. There are many ways to specify these parameters. If we use ZYX Euler angles, a general rotation is obtained by first rotating about the z axis by an angle θ, then rotating about the y axis by angle η, finally rotating about the x axis by angle ζ. The transformation matrix $[A]$ is obtained as follows:

$$[A] = \begin{bmatrix} c\theta & -s\theta & 0 \\ s\theta & c\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c\eta & 0 & s\eta \\ 0 & 1 & 0 \\ -s\eta & 0 & c\eta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\zeta & -s\zeta \\ 0 & s\zeta & c\zeta \end{bmatrix}$$

$$= \begin{bmatrix} c\theta c\eta & -s\theta c\zeta + c\theta s\eta s\zeta & s\theta s\zeta + c\theta s\eta c\zeta \\ s\theta c\eta & c\theta c\zeta + s\theta s\eta s\zeta & -c\theta s\zeta + s\theta s\eta c\zeta \\ -s\eta & c\eta s\zeta & c\eta c\zeta \end{bmatrix} \qquad (2\text{-}22)$$

where symbols s and c represent the sine and cosine functions.

2.3.2 Quaternions

The rotation matrix $[A]$ can also be parameterized by using Euler parameters which require only one single rotation about a rotation axis. For a rotation of angle ϕ about a unit vector $\boldsymbol{u} = (u_x, u_y, u_z)$, the Euler parameters of the rotation can be expressed as the set of four homogeneous parameters, q_1, q_2, q_3, and q_4 , called a *quaternion*, and given by

$$q_1 = u_x \sin(\phi/2),\ q_2 = u_y \sin(\phi/2),\ q_3 = u_z \sin(\phi/2),\ q_4 = \cos(\phi/2) \qquad (2\text{-}23)$$

The rotation angle ϕ and the components of the unit vector (u_x, u_y, u_z) can be obtained from the components, a_{ij}, $i, j = 1, 2, 3$ of $[A]$ by the following relationships:

$$\phi = \cos^{-1}\{ (a_{11} + a_{22} + a_{33} - 1)/2 \}, \qquad (2\text{-}24)$$

and

$$u_x = (a_{32} - a_{23})/(2 \sin\phi),$$

$$u_y = (a_{13} - a_{31})/(2 \sin\phi),$$

$$u_z = (a_{21} - a_{12})/(2 \sin\phi). \qquad (2\text{-}25)$$

The Euler parameters of this orthogonal matrix are used to define the mapping of a spherical displacement representing the spherical displacement onto a point in the space of mapping. Letting x, y, z, and w be the coordinates of a point in a four-dimensional space, we have

$$x = u_x \sin(\phi/2),$$

$$y = u_y \sin(\phi/2),$$

$$z = u_z \sin(\phi/2,$$

$$w = \cos(\phi/2).$$

which represent a rotation of angle ϕ about a unit axis $\boldsymbol{u} = (u_x, u_y, u_z)$. Therefore, this is also a unit quaternion because the sum of the squares of its components equals unity. In general, the rotation of an angle θ about the z-coordinate axis, $\boldsymbol{u} = (0, 0, 1)$, can now be written as $\boldsymbol{Q} = \cos(\theta/2) + \sin(\theta/2)\ \vec{k}$. Similarly, the rotation of an angle η about the y-coordinate axis is $\boldsymbol{Q} = \cos(\eta/2) + \sin(\eta/2)\ \vec{j}$ and the rotation of an angle ζ about the x-coordinate axis is $\boldsymbol{Q} = \cos(\zeta/2) + \sin(\zeta/2)\ \vec{i}$.

2.3.3 Quaternion Product

Successive spherical displacements can be expressed as the successive products of their corresponding quaternions. A quaternion $\boldsymbol{Q} = q_4 + q_1\ \vec{i} + q_2\ \vec{j} + q_3\ \vec{k}$ can also be written as a four dimensional vector form $\boldsymbol{Q} = (q_1, q_2, q_3, q_4)$. The product of two quaternions $\boldsymbol{Q} = (q_1, q_2, q_3, q_4)$ and $\boldsymbol{H} = (h_1, h_2, h_3, h_4)$ results in a product of their associated vectors, $\boldsymbol{QH}$:

$$\begin{aligned}\boldsymbol{QH} = {} & q_4h_4 - q_1h_1 - q_2h_2 - q_3h_3 \\ & + (q_4h_1 - q_3h_2 + q_2h_3 + q_1h_4)\ \vec{i} \\ & + (q_3h_1 + q_4h_2 - q_1h_3 + q_2h_4)\ \vec{j} \\ & + (q_4h_3 + q_3h_4 + q_1h_2 - q_2h_1)\ \vec{k}\end{aligned} \tag{2-27}$$

The components of $\boldsymbol{QH}$ can be obtained by representing the Clifford algebra in matrix form. From eq(2-27): $\boldsymbol{QH} = [Q^+]\boldsymbol{H}$,

$$[Q^+]\boldsymbol{H} = \begin{bmatrix} q_4 & -q_3 & q_2 & q_1 \\ q_3 & q_4 & -q_1 & q_2 \\ -q_2 & q_1 & q_4 & q_3 \\ -q_1 & -q_2 & -q_3 & q_4 \end{bmatrix} \begin{Bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \end{Bmatrix} \tag{2-28}$$

or $QH = [H^-]Q$

$$[H^-]Q = \begin{bmatrix} h_4 & h_3 & -h_2 & h_1 \\ -h_3 & h_4 & h_1 & h_2 \\ h_2 & -h_1 & h_4 & h_3 \\ -h_1 & -h_2 & -h_3 & h_4 \end{bmatrix} \begin{Bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{Bmatrix} \tag{2-29}$$

2.4 Spatial Displacements and Dual Quaternions

2.4.1 Spatial Displacement of a Rigid Body

In a three-dimensional Euclidean space, a general displacement of a rigid body is determined by six parameters. Unlike a planar displacement which has a fixed point called the pole of the displacement, a spatial displacement has a fixed line, called the screw axis, that has the same position in space before and after the displacement. The relative position of two rigid bodies in three dimensional space is defined by the transformation that specifies the coordinates $\boldsymbol{X}=(X, Y, Z)$ in the base frame F in terms of the coordinates $\boldsymbol{x} = (x, y, z)$ of a point in the moving body measured in its frame M (see Figure 2-3).

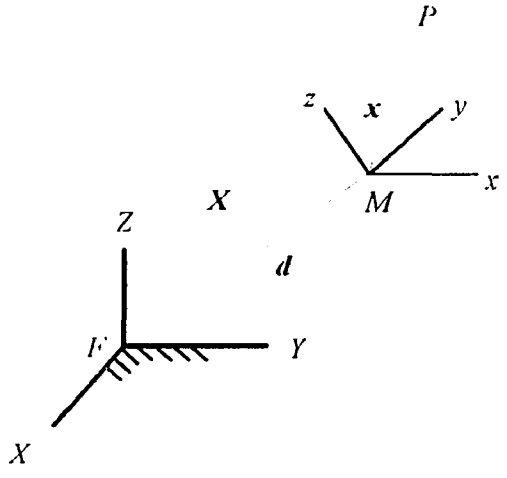


Figure 2-3 A general displacement of a rigid body

The transformation in three-dimensional space is given as

$$X = [A]x + d \tag{2-30}$$

which is analogous to the planar displacement of eq.(2-5). The rotation matrix $[A]$ defines the orientation of frame M relative to frame F and the translation of vector d gives the origin of frame M. The general displacement of a rigid body can be specified by combining a translation with a rotation.

2.4.2 Dual Quaternions

Finally, we discuss the mapping of a general displacement in physical space (Bottema and Roth, 1979, Ravani and Roth ,1982, and Ge and Ravani, 1991).

The transformation given in eq.(2-30) can be mapped into a higher dimension again by using the concept of Euler parameters. A general displacement of a rigid body can be determined by a screw displacement which is a rotation about and a translation along the screw axis. Let θ be the angle of rotation and d be the distance of translation. The screw displacement can be written as a dual angle

$$\hat{\theta} = \theta + \varepsilon d, \tag{2-31}$$

where ε is the dual number. The screw axis is defined by

$$\hat{u} = (\hat{u}_x, \hat{u}_y, \hat{u}_z) = (u_x + \varepsilon u_x^0, u_y + \varepsilon u_y^0, u_z + \varepsilon u_z^0) \tag{2-32}$$

which is a dual vector. The primary component $u = (u_x, u_y, u_z)$ is the direction vector of the screw axis and the dual component $u^0 = u_x^0 + u_y^0 + u_z^0$ is the moment of the vector u about the origin of the fixed frame. The Euler parameters of this screw displacement (dual Euler parameters) $\hat{Q} = Q + \varepsilon Q^0$, can also be assembled into a *dual quaternion.* The primary part, $Q = q_4 + \varepsilon q_1 \vec{i} + \varepsilon q_2 \vec{j} + q_3 \vec{k} = (q_1, q_2, q_3, q_4)$ is defined by the Euler parameter of the rotation matrix $[A]$. The dual part, $Q^0 = q_4^0 + q_1^0 \vec{i} + q_2^0 \vec{j} + q_3^0 \vec{k} = (q_1^0, q_2^0, q_3^0, q_4^0)$, is defined by the formula:

$$(Q^0)^T = \frac{1}{2}[d]Q^T \tag{2-33}$$

or

$$\begin{bmatrix} q_1^0 \\ q_2^0 \\ q_3^0 \\ q_4^0 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 0 & -d_3 & d_2 & d_1 \\ d_3 & 0 & -d_1 & d_2 \\ -d_2 & d_1 & 0 & d_3 \\ -d_1 & -d_2 & -d_3 & 0 \end{bmatrix}\begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix}$$

where appears the translation vector $\boldsymbol{d} = d_1\vec{i} + d_2\vec{j} + d_3\vec{k}$. Substituting the Euler parameters of the rotation matrix $[A]$ into $\hat{Q}$, the components of $\hat{Q}$ are written as

$$\hat{q}_1 = \hat{u}_x \sin(\hat{\theta}/2), \hat{q}_2 = \hat{u}_y \sin(\hat{\theta}/2), \hat{q}_3 = \hat{u}_z \sin(\hat{\theta}/2), \hat{q}_4 = \cos(\hat{\theta}/2) \quad (2\text{-}34)$$

where the dual angles

$$\sin(\hat{\theta}/2) = \sin(\theta/2) + \varepsilon(d/2)\cos(\theta/2), \quad (2\text{-}35)$$

$$\cos(\hat{\theta}/2) = \cos(\theta/2) - \varepsilon(d/2)\sin(\theta/2). \quad (2\text{-}36)$$

2.4.3 Dual Quaternion Product

The dual quaternion $\hat{Q} = Q + \varepsilon Q^0$, when written in vector form, becomes an eight dimensional vector $\hat{Q} = (Q, Q^0) \in \boldsymbol{R}^8$. The product of two dual quaternions $\hat{Q}\hat{H}$ can also be written in matrix forms (McCarthy, 1990):

$$\begin{Bmatrix} QH \\ (QH)^0 \end{Bmatrix} = \begin{bmatrix} [Q'] & [0] \\ [Q^{0'}] & [Q'] \end{bmatrix}\begin{bmatrix} H \\ H^0 \end{bmatrix} \quad (2\text{-}36)$$

or

$$\begin{Bmatrix} QH \\ (QH)^0 \end{Bmatrix} = \begin{bmatrix} [H'] & [0] \\ [H^{0-}] & [H'] \end{bmatrix}\begin{bmatrix} G \\ G^0 \end{bmatrix} \quad (2\text{-}37)$$

where the 4x4 matrices $[Q']$, $[Q^{0'}]$, $[H']$ and $[H^{0-}]$ are obtained from the matrix forms of quaternion products.

CHAPTER 3

CONSTRAINT MANIFOLDS FOR MOTION SYNTHESIS

3.1 Introduction

A mechanism consists of a sequence of links, generally considered rigid, which are connected by joints, such as revolute joints or ball joints, to form an open or closed chain. The position and orientation of the end link of an open chain relative to the base frame can be obtained by a sequence of coordinate transformations which define the position and orientation of each link relative to its neighbor in the chain from the base frame to the end link. Normally, this is done by using a sequence of matrix multiplications. By carrying out the multiplications to obtain an explicit form for the position and orientation of the end link, we obtained the structure equation of an open chain. Using quaternion algebra, the structure equation of an open chain mechanism forms the constraint manifold of the open chain in a higher-dimensional image space. The manifold can be a parameterized curve, surface, or hypersurface, depending on the degree of freedom of the chain. A constraint manifold defines all the reachable points of an open chain mechanism in the image space. It also represents the position and orientation of the end link constructed by the rest of the chain in the physical space. The constraint manifold of a closed chain is the intersection of the constraint manifolds of the structure equations of the two open chains which connect to form the complete mechanism.

The application of constraint manifolds for the synthesis of mechanisms in the image space is the main idea of this research. Papers dealing with the constraint manifolds and their applications to mechanisms include Ge and McCarthy (1989), Fischer (1990), McCarthy (1990), and others.

In this chapter we have developed the constraint manifolds suitable for the synthesis of a planar-four-bar mechanism, a spherical-four-bar mechanism, and a spatial four-bar mechanism.

3.2 Constraint Manifold of a Planar Four-Bar Mechanism

A general planar four-bar mechanism for motion synthesis is shown in Figure 3-1. The reference (fixed) frame is designated as F, and the moving frame, or the tracer frame, is designated as M. The coordinates of the two fixed pivots, o_1 and o_2, are (x_1, y_1) and (x_2, y_2) respectively. The lengths of the cranks are a_1 and a_2, and the dimensions of the coupler are denoted as b_1, b_2, and h. The angular orientation of the tracer frame M with respect to the reference frame F is denoted by angle η.

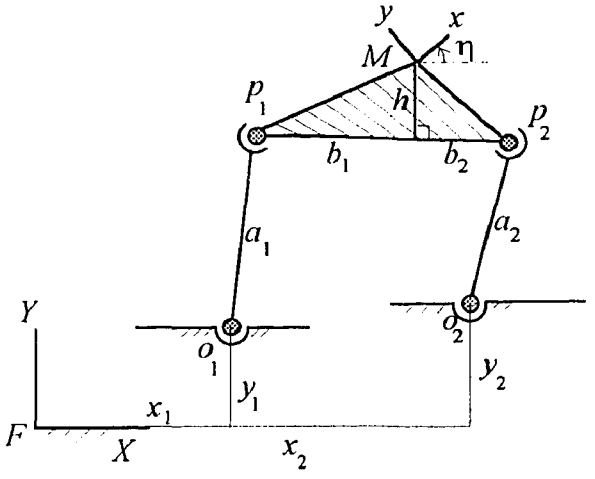


Figure 3-1 A general planar four-bar mechanism for motion synthesis.

Two structure equations can be formed representing the displacement of the tracer frame from the reference frame. One is obtained by the coordinate transformations through the left-hand side crank-coupler dyad, and the other is obtained similarly through the right-hand side crank-coupler dyad. Therefore, two constraint surfaces, each from one

of the structure equations, are obtained. The constraint manifold of a planar four-bar mechanism is a curve formed by the intersection of the constraint surfaces of the two structure equations.

3.2.1 Constraint Manifold of a 2R Open Chain

The constraint surface to be used for the motion synthesis problem is based on the crank-coupler dyad, a two-revolute joint (2R) open chain (Figure 3-2). The structure equation of 2R open chain is derived by a rotation about joint axis o by the displacement angle θ, a translation along the link by the distance a, and a rotation about joint axis p by the displacement angle ϕ.

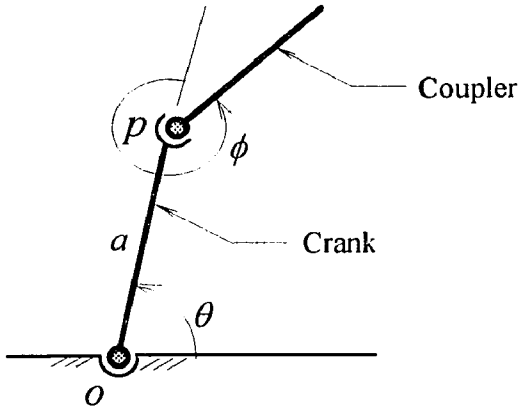


Figure 3-2 A 2R crank-coupler dyad with link a and joints o and p.

In term of matrix transformation this becomes

$$[S] = [Z(\theta)][X(a)][Z[(\phi)]$$

or

$$[S] = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & a \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3-1}$$

which can also be written as a product of planar quaternions:

$$S = Z(\theta)X(a)Z(\phi)$$

or

$$S = (0, 0, \sin\theta/2, \cos\theta/2)(a/2, 0, 0, 1)(0, 0, \sin\phi/2, \cos\phi/2) \quad (3\text{-}2)$$

Using planar quaternion algebra, this product can be expanded as a parameterized surface in $\boldsymbol{R}^4$:

$$S(\theta, \phi) = (S_1(\theta, \phi, a), S_2(\theta, \phi, a), S_3(\theta, \phi), S_4(\theta, \phi)) \quad (3\text{-}3)$$

where

$$S_1(\theta, \phi) = \frac{a}{2}\cos(\frac{\theta-\phi}{2}),$$

$$S_2(\theta, \phi) = \frac{a}{2}\sin(\frac{\theta-\phi}{2}),$$

$$S_3(\theta, \phi) = \sin(\frac{\theta+\phi}{2}),$$

$$S_4(\theta, \phi) = \cos(\frac{\theta+\phi}{2}).$$

This is the constraint manifold of a 2R crank-coupler dyad. It represents the reachable points of the open chain in the image space in terms of the length of the link and the rotation angles of the joint. Eliminating the rotation angles θ and ϕ we have:

$$S_1^2 + S_2^2 - (\frac{a}{2})^2 S_3^2 - (\frac{a}{2})^2 S_4^2 = 0 \quad (3\text{-}4)$$

Let positions $\boldsymbol{S} = (S_1, S_2, S_3, S_4)$ reachable by the end link of the 2R open chain be represented by a vector $X = (x, y, z, w)$. Eq.(3-4) defines a surface given by the equation

$$x^2 + y^2 - (\frac{a}{2})^2 z^2 - (\frac{a}{2})^2 w^2 = 0 \quad (3\text{-}5)$$

It has been shown (McCarthy, 1990) that this equation can be written as the quadratic form

$$\boldsymbol{X}^T[Q]\boldsymbol{X} = 0 \quad (3\text{-}6)$$

where the 4 x 4 coefficient matrix has the form

$$[Q] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{a^2}{4} & 0 \\ 0 & 0 & 0 & -\frac{a^2}{4} \end{bmatrix} \tag{3-7}$$

The 2R constraint manifold can be visualized by projecting on the $w = 1$ plane which produces a right circular hyperboloid of one sheet (Figure 3-3). This hyperboloid is centered at the point $x = y = z = 0$ and is aligned with the z-axis and has the minimum diameter a at the center circle.

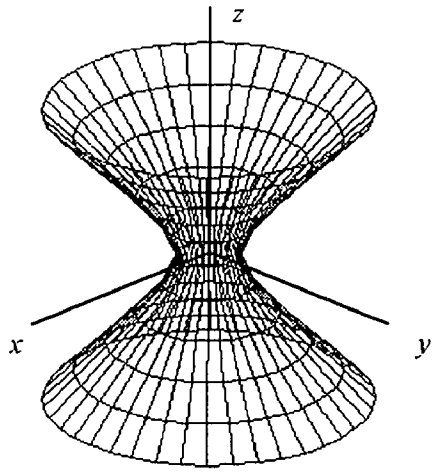


Figure 3-3 Constraint manifold of a 2R open chain.

The canonical form of eq.(3-7) is the result of the choice of fixed and moving frame. For motion synthesis of mechsnisms, we have to move the moving frame from the joint p to the tracer frame and move the fixed frame from the joint o to the global reference frame.

3.2.2 Constraint Manifold of the Left-Hand Side Crank-Coupler Dyad

The constraint manifold of the left-hand side crank-coupler dyad (Figure 3-4) is developed by considering the transformation from the reference frame through a distance (x_1, y_1) to the joint o_1 of the link a_1, through the link length a_1 and into the tracer frame through a distance (b_1, h) and a rotation of angle η.

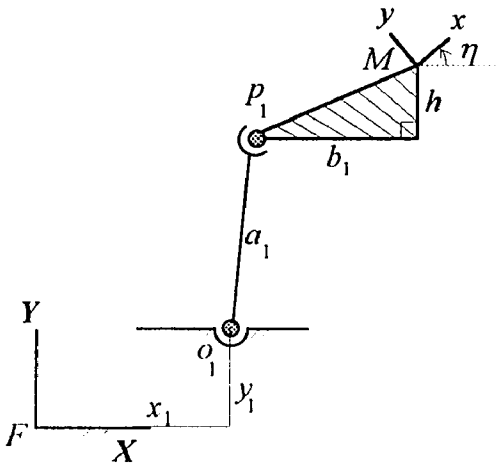


Figure 3-4 The left-hand side crank-coupler dyad.

In terms of quaternion product, this can be expressed as a premultiplication of displacement (x_1, y_1) with eq.(3-3), followed by a postmultiplications of a displacement (b_1, h) and a rotation of angle η. Therefore, we have

$$S_L = (x_1, y_1, 0,1)\, S(\theta, \phi)\, (b_1, h, 0, 1)\, (0,0,\sin\eta/2, \cos\eta/2) \tag{3-8}$$

Instead of carrying out the quaternion multiplication, S_L is written as a linear transformation of the vector $S(\theta, \phi)$ given in eq.(3-3):

$$S_L = [C_1]S \tag{3-9}$$

where $[C_1]$ can be obtained by multiplications of matrices as shown in eqs.(2-19, 2-20) so that

$$[C_l] = \begin{bmatrix} 1 & 0 & y_1 & x_1 \\ 0 & 1 & -x_1 & y_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -h & b_1 \\ 0 & 1 & b_1 & h \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c\eta & s\eta & 0 & 0 \\ -s\eta & c\eta & 0 & 0 \\ 0 & 0 & c\eta & s\eta \\ 0 & 0 & -s\eta & c\eta \end{bmatrix}$$

$$= \begin{bmatrix} c\eta & s\eta & \dfrac{-hc\eta + y_1 c\eta + b_1 s\eta - x_1 s\eta}{2} & \dfrac{b_1 c\eta + x_1 c\eta + hs\eta + y_1 s\eta}{2} \\ -s\eta & c\eta & \dfrac{b_1 c\eta - x_1 c\eta + hs\eta - y_1 s\eta}{2} & \dfrac{hc\eta + y_1 c\eta - b_1 s\eta - x_1 s\eta}{2} \\ 0 & 0 & c\eta & s\eta \\ 0 & 0 & -s\eta & c\eta \end{bmatrix} \tag{3-10}$$

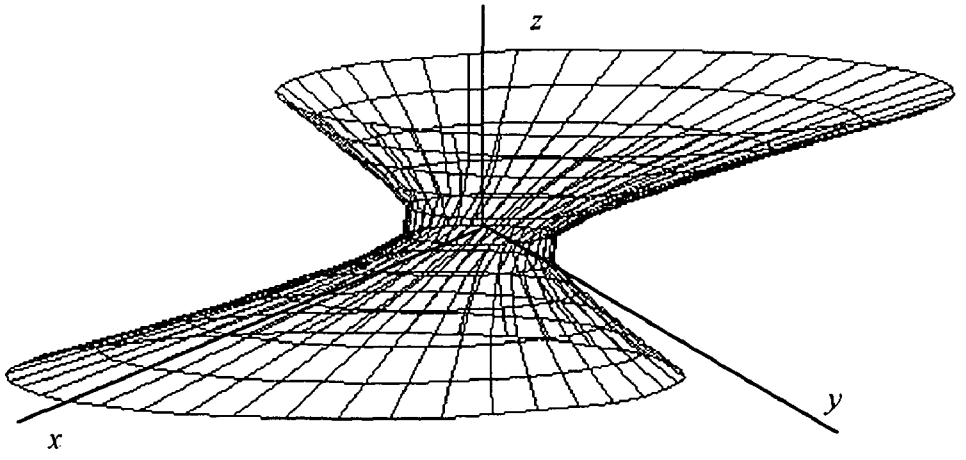


Figure 3-5 The constraint manifold after a linear transformation.

There exists a constraint manifold equation similar to eq.(3-5) based on quaternion product S_L. The constraint manifold that also satisfies eq.(2-10) represents the reachable points of the left-hand side crank-coupler open chain in the image space. When projecting on the $w = 1$ plane, the constraint manifold becomes a skew hyperboloid of one sheet (Figure 3-5). Written in quadratic form, it becomes

$$X_1^T [Q_1] X_1 = 0 \tag{3-11}$$

where vector X_1 is a linear transformation of vector **X** such that

$$X_1 = [C_1]\mathbf{X} \tag{3-12}$$

Substituting eq.(3-12) into eq.(3-11), we have

$$([C_1]^{-1}\mathbf{X})^T[Q]([C_1]^{-1}\mathbf{X}) = 0 \tag{3-13}$$

which can be written as

$$X^T[Q_1]X = 0 \tag{3-14}$$

where

$$[Q_1] = ([C_1]^{-1})^T[Q]([C_1]^{-1})$$

$$= \begin{bmatrix} 1 & 0 & \frac{\alpha_1 - y_1}{2} & \frac{-(\beta_1 + x_1)}{2} \\ 0 & 1 & \frac{x_1 - \beta_1}{2} & \frac{-(\alpha_1 + y_1)}{2} \\ \frac{\alpha_1 - y_1}{2} & \frac{x_1 - \beta_1}{2} & \rho_1 - \frac{\sigma_1}{2} & \frac{\gamma_1}{2} \\ \frac{-(\beta_1 + x_1)}{2} & \frac{-(\alpha_1 + y_1)}{2} & \frac{\gamma_1}{2} & \rho_1 + \frac{\sigma_1}{2} \end{bmatrix} \tag{3-15}$$

and

$$\alpha_1 = h\cos\eta - b_1\sin\eta,$$

$$\beta_1 = b_1\cos\eta + h\sin\eta,$$

$$\gamma_1 = -h\,x_1\cos\eta + b_1\,y_1\cos\eta + b_1\,x_1\sin\eta + h\,y_1\sin\eta,$$

$$\rho_1 = (-a_1^2 + b_1^2 + h^2 + x_1^2 + y_1^2)/4,$$

$$\sigma_1 = b_1\,x_1\cos\eta + h\,y_1\cos\eta + h\,x_1\sin\eta - b_1\,y_1\sin\eta.$$

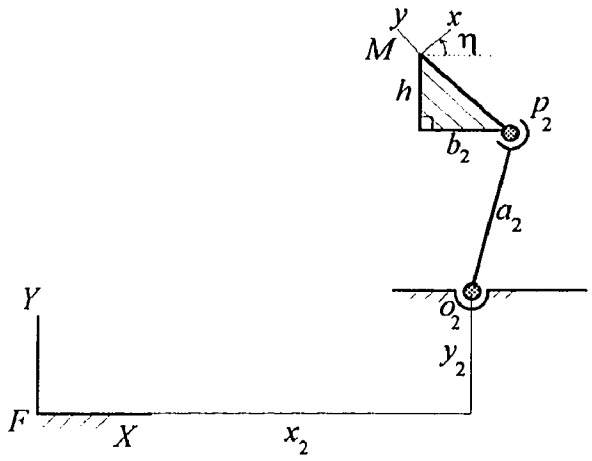


Figure 3-6 The right-hand side crank-coupler dyad.

3.2.3 Constraint Manifold of the Right-Hand Side Crank-Coupler Dyad

The constraint manifold of the right-hand side crank-coupler dyad (Figure 3-6) is developed by considering the coordinate transformations from the reference frame through a distance (x_2, y_2) to the joint o_2 of the dyad, then through the link length a_2 and into the tracer frame through a distance $(-b_2, h)$ and a rotation of angle η.

In terms of quaternion products, this can be expressed as a premultiplication of displacement (x_2, y_2) with eq.(3-3), then postmultiplications of a displacement $(-b_2, h)$ and a rotation of angle η. Therefore, we have

$$S_2 = (x_2, y_2, 0, 1)\, S(\theta, \phi)\, (-b_2, h, 0, 1)\, (0, 0, \sin\eta/2, \cos\eta/2) \tag{3-16}$$

Similarly,

$$S_2 = [C_2]S \tag{3-17}$$

where

$$[C_2] = \begin{bmatrix} c\eta & s\eta & \dfrac{-hc\eta + y_2 c\eta - b_2 s\eta - x_2 s\eta}{2} & \dfrac{-b_2 c\eta + x_2 c\eta + hs\eta + y_2 s\eta}{2} \\ -s\eta & c\eta & \dfrac{-b_2 c\eta - x_2 c\eta + hs\eta - y_2 s\eta}{2} & \dfrac{hc\eta + y_2 c\eta + b_2 s\eta - x_2 s\eta}{2} \\ 0 & 0 & c\eta & s\eta \\ 0 & 0 & -s\eta & c\eta \end{bmatrix} \quad (3\text{-}18)$$

Again, there exists a constraint manifold equation similar to eq.(3-5) based on quaternion product S_R. Written in quadratic form, it becomes

$$\boldsymbol{X}^T[Q_2]\boldsymbol{X} = 0 \quad (3\text{-}19)$$

The coefficient matrix is

$$[Q_2] = ([C_2]^{-1})^T[Q]([C_2]^{-1})$$

$$= \begin{bmatrix} 1 & 0 & \dfrac{\alpha_2 - y_2}{2} & \dfrac{\beta_2 - x_2}{2} \\ 0 & 1 & \dfrac{\beta_2 + x_2}{2} & \dfrac{-(\alpha_2 + y_2)}{2} \\ \dfrac{\alpha_2 - y_2}{2} & \dfrac{\beta_2 + x_2}{2} & \rho_2 + \dfrac{\sigma_2}{2} & \dfrac{\gamma_2}{2} \\ \dfrac{\beta_2 - x_2}{2} & \dfrac{-(\alpha_2 + y_2)}{2} & \dfrac{\gamma_2}{2} & \rho_2 - \dfrac{\sigma_2}{2} \end{bmatrix} \quad (3\text{-}20)$$

where

$$\alpha_2 = h\cos\eta + b_2\sin\eta,$$

$$\beta_2 = b_2\cos\eta - h\sin\eta,$$

$$\gamma_2 = -h x_2\cos\eta - b_2 y_2\cos\eta - b_2 x_2\sin\eta + h y_2\sin\eta,$$

$$\rho_2 = (-a_2^2 + b_2^2 + h^2 + x_2^2 + y_2^2)/4,$$

$$\sigma_2 = b_2 x_2\cos\eta - h y_2\cos\eta - h x_2\sin\eta - b_2 y_2\sin\eta.$$

Eq.(3-19) represents all the reachable points of the right-hand side crank-coupler open chain in the image space. Again, when projecting on the $w = 1$ plane, the constraint manifold becomes a skew hyperboloid of one sheet. The intersection of the two constraint manifolds eq.(3-5) and eq.(3-19) defines the image curve through which the coupler of the four-bar linkage will pass (Figure 3-7). Furthermore, the image curve must also satisfy eq.(2-15), the constraint equation of the planar quaternion. Therefore, we have a total of three equations in the four-dimensional image space resulting in one degree of freedom for planar four-bar mechanisms.

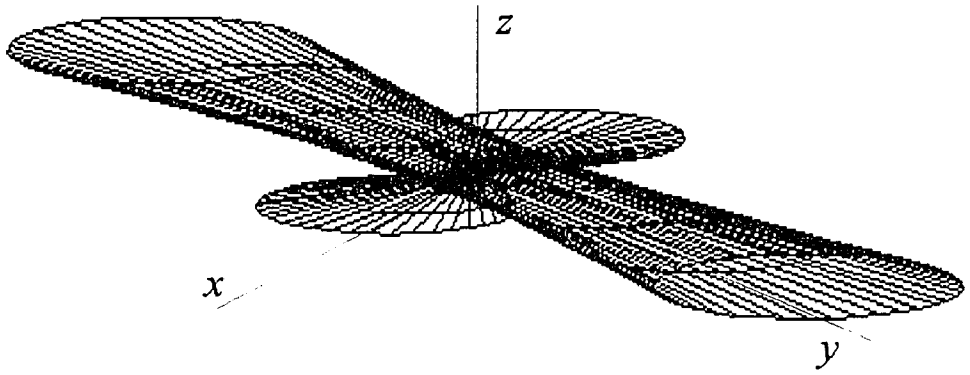


Figure 3-7 The intersection of two constraint manifolds.

3.3 Constraint Manifold of a Spherical Four-Bar Mechanism

A spherical four-bar mechanism is a special type of four-bar mechanism with the property that the axes of the joints all intersect at a single point, c (Figure 3-8). By fixing one of the links, we obtain a single degree of freedom spatial mechanism similar to a planar four-bar mechanism. For motion generation, the tracer frame is to pass through frames lying on a concentric sphere about the point c. The constraint manifold of a spherical four-bar is based on a 2R spherical crank-coupler dyad open chain shown in the next section.

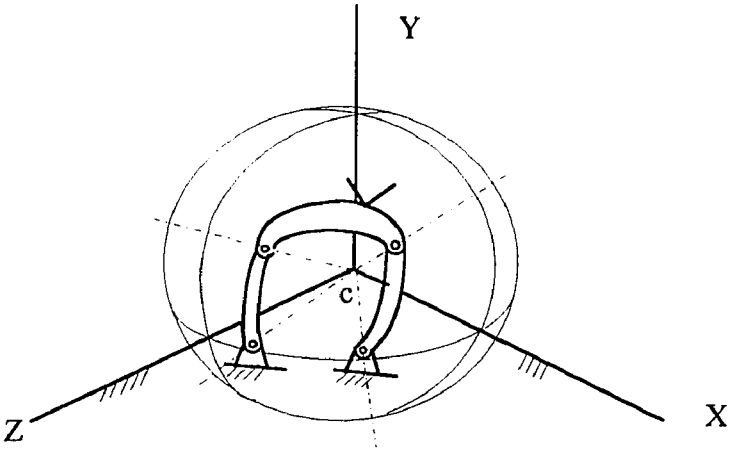


Figure 3-8 A spherical four-bar mechanism.

3.3.1 Constraint Manifold of a 2R Spherical Open Chain

A 2R spherical open chain, similar to a 2R planar open chain, has two revolute joints, but the axes of the joints intersect at the center point of a sphere at c, making an intersection angle β, and intersecting the sphere at points o and p (Figure 3-9). Let us assign frames $\boldsymbol{O}$ and $\boldsymbol{P}$ to have their origin at center c and their z-axes passing through o and p respectively.

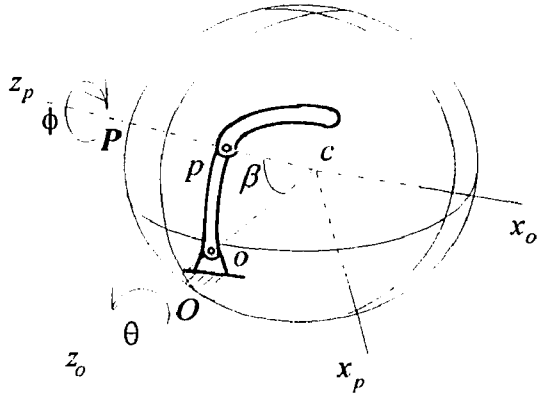


Figure 3-9 A 2R spherical open chain.

To obtain the constraint manifold of a 2R spherical open chain, let the rotation angle of the frame $\boldsymbol{O}$ about its z-axis be θ such that its x-axis is perpendicular to the plane defined by the z-axes and let the rotation of the frame $\boldsymbol{P}$ about its z-axis be angle ϕ. The

structure equation of the spherical crank-coupler dyad from the fixed pivot to the moving pivot is then defined by a rotation about the z-axis by the angle θ, followed by a rotation about the x-axis through the intersection angle β, and then a rotation about the z-axis through angle ϕ (Figure 3-9).

The quaternion form of this structure equation become

$$S = Z(\theta)X(\beta)Z(\phi) \tag{3-21}$$

or

$$S = (0, 0, \sin\theta/2, \cos\theta/2)(\sin\beta/2, 0, 0, \cos\beta/2)(0, 0, \sin\phi/2, \cos\phi/2)$$

Expanding the quaternion product we have the parameterized surface

$$S(\beta, \theta, \phi) = (S_1(\beta, \theta, \phi), S_2(\beta, \theta, \phi), S_3(\beta, \theta, \phi), S_4(\beta, \theta, \phi)) \tag{3-22}$$

where

$$S_1(\beta, \theta, \phi) = \sin(\frac{\beta}{2}) \cos(\frac{\theta-\phi}{2}),$$

$$S_2(\beta, \theta, \phi) = \sin(\frac{\beta}{2}) \sin(\frac{\theta-\phi}{2}),$$

$$S_3(\beta, \theta, \phi) = \cos(\frac{\beta}{2}) \sin(\frac{\theta+\phi}{2}),$$

$$S_4(\beta, \theta, \phi) = \cos(\frac{\beta}{2}) \cos(\frac{\theta+\phi}{2}).$$

Angles θ and ϕ in eq.(3-19) can be eliminated to obtain the algebraic equation parameterized by the intersection angle β:

$$\cos^2(\frac{\beta}{2})S_1^2 + \cos^2(\frac{\beta}{2})S_2^2 - \sin^2(\frac{\beta}{2})S_3^2 - \sin^2(\frac{\beta}{2})S_4^2 = 0 \tag{3-23}$$

This is the constraint manifold of a 2R spherical crank-coupler dyad open chain. Let positions $\boldsymbol{S} = (S_1, S_2, S_3, S_4)$ reachable by the end link of the 2R open chain in the image space be represented by a vector $\boldsymbol{X} = (x, y, z, w)$. Eq.(3-23) defines a surface given by the equation:

$$\cos^2(\frac{\beta}{2})x^2 + \cos^2(\frac{\beta}{2})y^2 - \sin^2(\frac{\beta}{2})z^2 - \sin^2(\frac{\beta}{2})w^2 = 0 \tag{3-24}$$

This equation can be written as the quadratic form (McCarthy, 1990):

$$X^T[Q]X = 0 \tag{3-25}$$

where the 4 x 4 coefficient matrix

$$[Q] = \begin{bmatrix} \cos^2(\beta/2) & 0 & 0 & 0 \\ 0 & \cos^2(\beta/2) & 0 & 0 \\ 0 & 0 & -\sin^2(\beta/2) & 0 \\ 0 & 0 & 0 & -\sin^2(\beta/2) \end{bmatrix} \tag{3-26}$$

If we divide Eq.(3-24) by $\cos^2(\frac{\beta}{2})$ and project it on the $w = 1$ plane, we have

$$x^2 + y^2 - \tan^2(\frac{\beta}{2})z^2 = \tan^2(\frac{\beta}{2}) \tag{3-27}$$

which again is a right circular hyperboloid of one sheet. This hyperboloid is aligned with the z-axis and centered at the point $x = y = z = 0$. The radius of its center circle is $\tan(\frac{\beta}{2})$.

3.3.2 Constraint Manifold of the Left-Hand Side Crank-Coupler Dyad

The constraint manifold of the left-hand side crank-coupler dyad (Figure 3-10) is developed by considering the transformation from the reference frame through a rotation of angle ψ_1 about the Z-axis (symbolized as $R_Z(\psi_1)$) such that the X-axis is perpendicular to the plane defined by the Z-axis and the vector o_1-c followed by a rotation through angle α_1 about the X-axis to frame O_1 ($R_X(\alpha_1)$). The transformation from the fixed pivot to the moving pivot is through a link of twist β_1. It can be expressed by the 2R open chain in canonical form. Finally, the transformation from the fixed pivot to the tracer frame is through a rotation of angle γ_1 about the x-axis of frame P_1 ($R_X(\gamma_1)$), followed by a rotation of angle σ about the y-axis ($R_Y(\sigma)$), and a rotation of angle η about the z-axis ($R_Z(\eta)$).

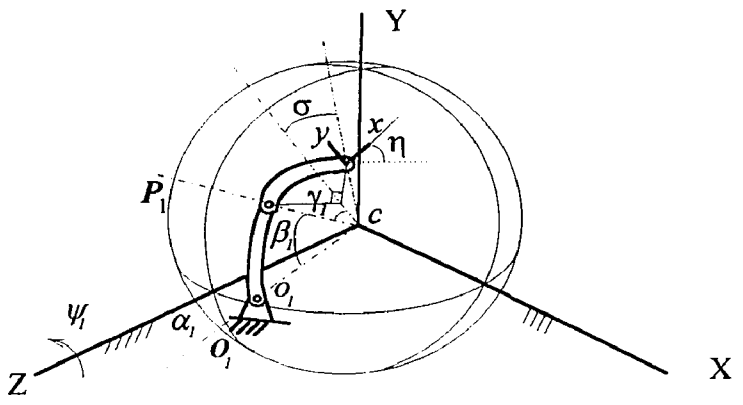


Figure 3-10 The spherical left-hand side crank-coupler dyad.

In terms of quaternion products, we have

$$S_L = S_1 \, S \, S_2 \tag{3-28}$$

where

$$S_1 = (0, 0, \sin\psi_1/2, \cos\psi_1/2)(\sin\alpha_1/2, 0, 0, \cos\alpha_1/2)$$

$$S_2 = (\sin\gamma_1/2, 0, 0, \cos\gamma_1/2)(0, \sin\sigma_1/2, 0, \cos\sigma_1/2)(0, 0, \sin\mu_1/2, \cos\mu_1/2).$$

Eq.(3-28) can also be written as a linear transformation of the vector $S(\alpha, \theta, \phi)$ given in eq.(3-21):

$$S_L = [C_1]S \tag{3-29}$$

In terms of symbolized notations, we have

$$[C_1] = [R_Z(\psi_1)^+]R_X[(\alpha_1)^+]R_Z[(\mu)^-]R_y[(\sigma)^-][R_X[(\gamma_1)^-] \tag{3-30}$$

Again, there exists a constraint manifold equation similar to eq.(3-25) based on quaternion product S_L. Written in quadratic form, it becomes

$$X_1^T[Q_1]X_1 = 0 \tag{3-31}$$

where vector X_1 is a linear transformation of vector X such that

$$X_1 = [C_1]X \tag{3-32}$$

This transforms the coefficient matrix $[Q]$ of eq(3-31) to

$$[Q_1] = ([C_1]^{-1})^T[Q]([C_1]^{-1}) \tag{3-33}$$

All the matrices in eq.(3-30) are orthogonal transformations, therefore $[C_1]$ is an orthogonal matrix and $[C_1]^{-1} = [C_1]^T$, so

$$[Q_1] = [C_1][Q]([C_1])^T \tag{3-34}$$

3.3.3 Constraint Manifold of the Right-Hand Side Crank-Coupler Dyad

The constraint manifold of the right-hand side crank-coupler dyad (Figure 3-11) is developed by considering the transformation from the reference frame through a rotation of angle ψ_2 about the Z-axis (symbolized as $R_z(\psi_2)$) such that the X-axis is perpendicular to the plane defined by the Z-axis and the vector o_2-c, followed by a rotation α_2 about the X-axis to frame O_2 ($R_X(\alpha_2)$). The transformation from the fixed pivot to the moving pivot is through a link of twist β_2. Finally, the transformation from the fixed pivot to the tracer frame is through a rotation of angle $(-\gamma_2)$ about the x-axis of frame P_2 ($R_X(-\gamma_2)$), followed by a rotation σ about the y-axis ($R_y(\sigma)$), and a rotation η about the z-axis ($R_z(\eta)$).

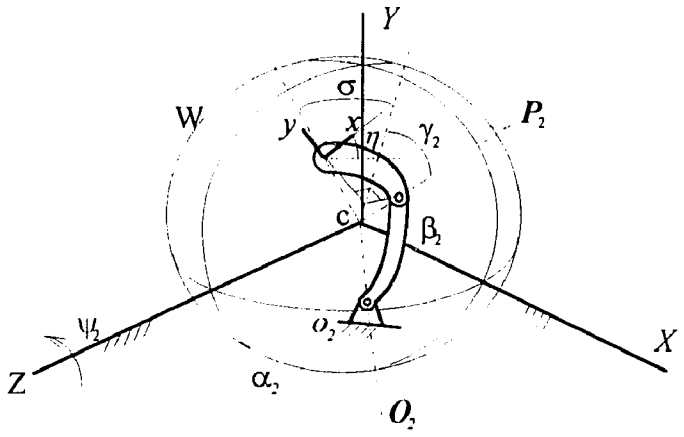


Figure 3-11 The right-hand side crank-coupler dyad.

In terms of quaternion products, we have

$$S_R = S_1 \, S \, S_2 \tag{3-35}$$

where

$$S_1 = (0, 0, \sin\psi_2/2, \cos\psi_2/2)(\sin\alpha_2/2, 0, 0, \cos\alpha_2/2)$$

$$S_2 = (-\sin\gamma_2/2, 0, 0, \cos\gamma_2/2)(0, \sin\sigma/2, 0, \cos\sigma/2)(0, 0, \sin\mu/2, \cos\mu/2).$$

S_R can be written as a linear transformation of the vector $S(\alpha, \theta, \phi)$ given in eq.(3-19):

$$S_R = [C_2]S \tag{3-36}$$

In terms of symbolized notations and matrix form of quaternions, we have

$$[C_2] = [R_Z(\psi_2)^+]R_X[(\alpha_2)^+]R_Z[(\mu)^-]R_y[(\sigma)^-][R_X[(-\gamma_2)^-] \tag{3-37}$$

The constraint manifold equation similar to eq.(3-19) based on quaternion product S_R. Written in quadratic form, this equation becomes

$$X_2^T[Q_2]X_2 = 0 \tag{3-38}$$

where vector X_2 is a linear transformation of vector **X** such that

$$X_2 = [C_2]\mathbf{X} \tag{3-39}$$

This transforms the coefficient matrix $[Q]$ of eq.(3-38) to

$$[Q_2] = ([C_2]^{-1})^T[Q]([C_2]^{-1}) \tag{3-40}$$

Again, all the matrices in eq.(3-37) are orthogonal transformations, therefore $[C_2]$ is an orthogonal matrix and $[C_2]^{-1} = [C_2]^T$, so

$$[Q_2] = [C_2][Q][C_2]^T \tag{3-41}$$

3.4 Constraint Manifold of a Spatial Four-Bar Mechanism

We next consider the constraint manifolds of three-dimensional mechanisms. A general spatial four-bar mechanism consists of four cylindrical joints which allow two different motions, rotation and translation, between two links (Figure 3-12). In general, the four axes of the spatial mechanism are not parallel with each other nor do they intersect at a point. If the four axes of the mechanism were parallel with each other, the mechanism would be a planar four-bar mechanism. If the four axes of the mechanism intersected at a common point, the mechanism then would be a spherical four-bar mechanism.

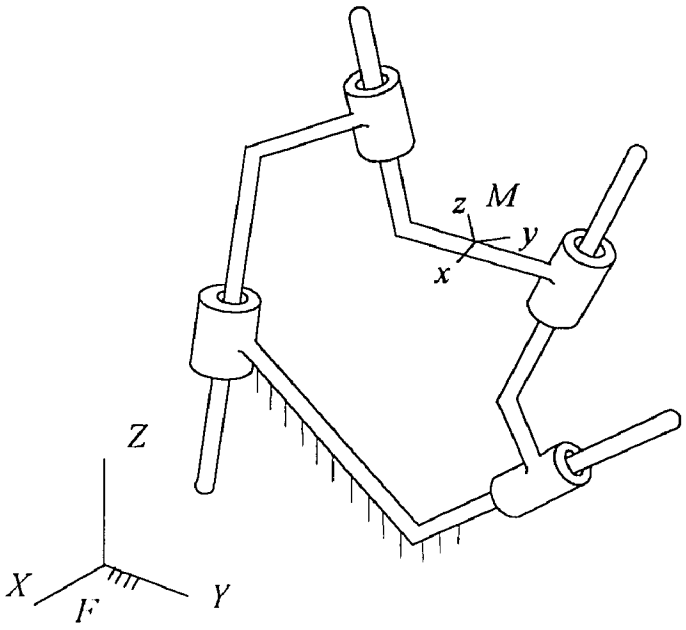


Figure 3-12 A general spatial four-bar mechanism.

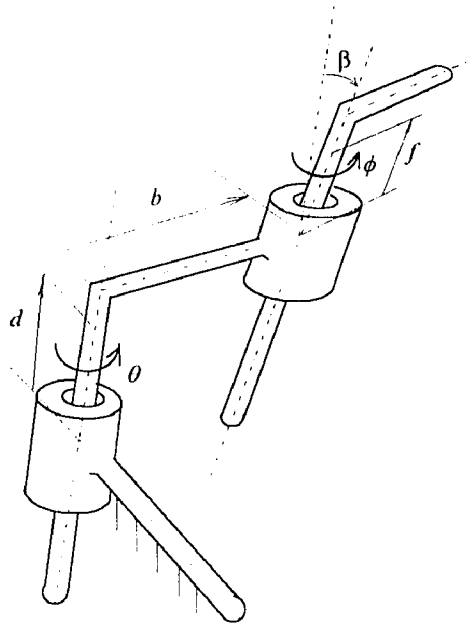


Figure 3-13 A 2C spatial open chain

3.4.1 Constraint Manifold of a 2C Spatial Open Chain

The constraint manifold of a spatial four-bar mechanism is based on a two-cylindrical joint (2C) open chain (Figure 3-13). The structure equation of a 2C open chain is obtained by a screw displacement through a dual angle $\hat{\theta} = \theta + \varepsilon d$ with respect to a z-axis aligned with the base joint, followed by a screw displacement of $\hat{\beta} = \beta + \varepsilon b$ along the x-axis aligned with the length of the link, and a screw displacement with respect to the axis of the joint with the coupler through a dual angle $\hat{\phi} = \phi + \varepsilon f$, which can be written as a product of dual quaternions:

$$S = Z(\hat{\theta})X(\hat{\beta})Z(\hat{\phi}) \tag{3-42}$$

or

$$S = (0, 0, \sin\hat{\theta}/2, \cos\hat{\theta}/2)(\sin\hat{\beta}/2, 0, 0, \cos\hat{\beta}/2)(0, 0, \sin\hat{\phi}/2, \cos\hat{\phi}/2)$$

Expanding the quaternion product we have the parameterized surface

$$S(\hat{\beta}, \hat{\theta}, \hat{\phi}) = (S_1(\hat{\beta}, \hat{\theta}, \hat{\phi}), S_2(\hat{\beta}, \hat{\theta}, \hat{\phi}), S_3(\hat{\beta}, \hat{\theta}, \hat{\phi}), S_4(\hat{\beta}, \hat{\theta}, \hat{\phi})) \tag{3-43}$$

where

$$S_1(\hat{\beta}, \hat{\theta}, \hat{\phi}) = \sin(\frac{\hat{\beta}}{2}) \cos(\frac{\hat{\theta}-\hat{\phi}}{2}),$$

$$S_2(\hat{\beta}, \hat{\theta}, \hat{\phi}) = \sin(\frac{\hat{\beta}}{2}) \sin(\frac{\hat{\theta}-\hat{\phi}}{2}),$$

$$S_3(\hat{\beta}, \hat{\theta}, \hat{\phi}) = \cos(\frac{\hat{\beta}}{2}) \sin(\frac{\hat{\theta}+\hat{\phi}}{2}),$$

$$S_4(\hat{\beta}, \hat{\theta}, \hat{\phi}) = \cos(\frac{\hat{\beta}}{2}) \cos(\frac{\hat{\theta}+\hat{\phi}}{2}).$$

Angles $\hat{\theta}$ and $\hat{\phi}$ in eq.(3-43) can be eliminated to obtain the algebraic form of the constraint manifold parameterized by angle $\hat{\beta}$:

$$\cos^2(\frac{\hat{\beta}}{2})S_1^2 + \cos^2(\frac{\hat{\beta}}{2})S_2^2 - \sin^2(\frac{\hat{\beta}}{2})S_3^2 - \sin^2(\frac{\hat{\beta}}{2})S_4^2 = 0 \tag{3-44}$$

This equation can be written as the quadratic form

$$\hat{X}[\hat{Q}]\hat{X}^T = 0 \tag{3-45}$$

where the points in the image space are represented by the vector

$$\hat{X} = (\hat{x}, \hat{y}, \hat{z}, \hat{w}) \tag{3-46}$$

and the 4 x 4 coefficient matrix

$$[\hat{Q}] = \begin{bmatrix} \cos^2(\hat{\beta}/2) & 0 & 0 & 0 \\ 0 & \cos^2(\hat{\beta}/2) & 0 & 0 \\ 0 & 0 & -\sin^2(\hat{\beta}/2) & 0 \\ 0 & 0 & 0 & -\sin^2(\hat{\beta}/2) \end{bmatrix} \tag{3-47}$$

Again, the canonical form of eq.(3-47) is the result of the choice of fixed and moving frame. For motion synthesis of mechanisms, we have to move the moving frame from the moving joint to the tracer frame and move the fixed frame from the fixed joint to the global reference frame. This can be done by linear transformations of eq.(3-47). Therefore, two dual constraint manifolds can be obtained by the dual quaternion operators from the reference frame through each leg to the tracer frame. We also have two constraint equations associated with the dual quaternion space:

$$X_1X_1^0 + X_2X_2^0 + X_3X_3^0 + X_4X_4^0 = 0 \tag{3-48}$$

and

$$X_1^2 + X_2^2 + X_3^2 + X_4^2 - 1 = 0 \tag{3-49}$$

Eq.(3-48) comes from the definition of the dual part of dual quaternions (eq.(2-33)) and eq.(3-49) is the constraint equation for the normalized coordinates.

Since each dual equation can be separated into two equations, we have a total of six equations in the eight-dimensional image space resulting in two degrees of freedom (DOF) for general three-dimensional spatial four-bar mechanisms. If we fixed one rotation or one translation in any joint of the spatial four-bar mechanism, one more constraint equation will be needed to reduce the motion to only one DOF. The method

applied to the general spatial four-bar mechanisms can also be applied to more complicated spatial mechanisms such as the RSSR-SS mechanism shown in Figure 3-14. Three eight-dimensional constraint equations can be obtained by the coordinate transformations from the reference frame to the tracer frame through the three legs and additional two four-dimensional constraint manifolds can also be obtained by planar quaternions through the two revolute-joint legs. A total of seven equations, including eqs.(3-48) and (3-49), can be formulated in the eight-dimensional image space resulting in a one DOF motion for the RSSR-SS spatial mechanism.

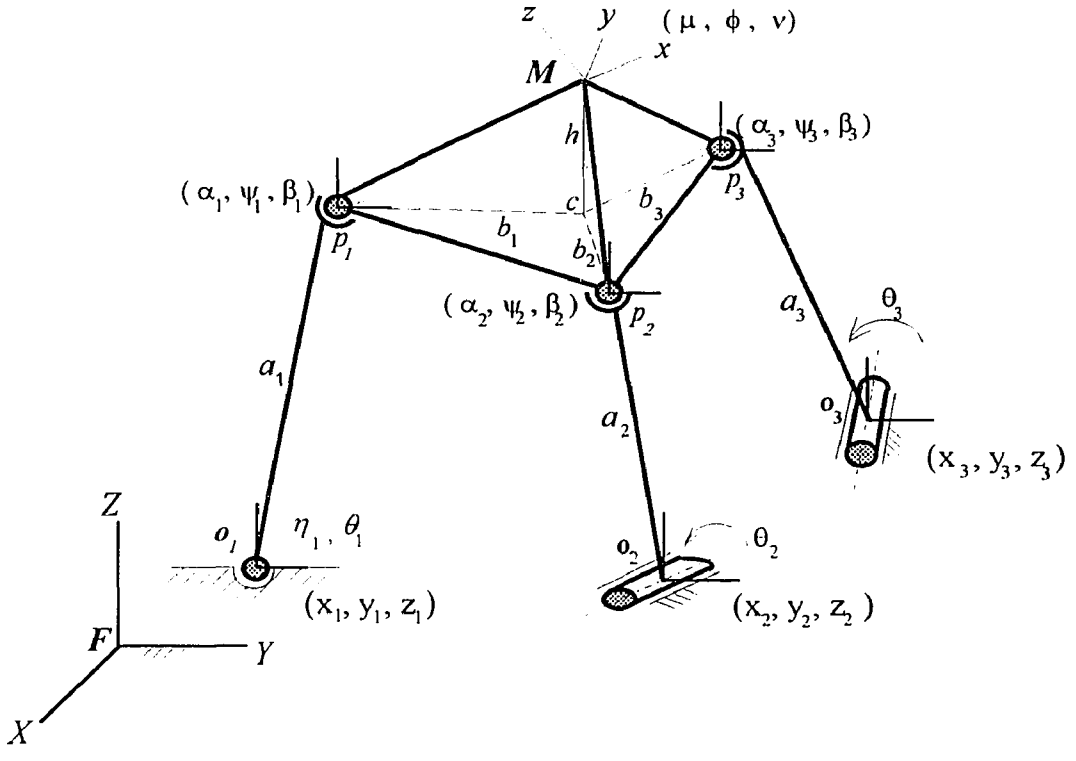


Figure 3-14 A RSSR-SS mechanism.

CHAPTER 4

CURVE FITTINGS IN IMAGE SPACE

4.1 Introduction

By means of kinematic mapping, the prescribed situations of the tracer frame of a mechanism can be mapped onto points. The motion of the tracer frame becomes a curve in the mapped space. Therefore, the problem of motion synthesis becomes one of finding a curve which passes through, or as close as possible to, the prescribed points. Although a space curve can be represented in many ways, it is convenient for us to use the constraint manifold equations derived in the last chapter for the motion synthesis problems. The image space curve can be obtained by the intersection of the constraint surfaces represented by the constraint manifold equations. It is known that when the total number of prescribed positions is five or less, an image curve can be found to pass through these prescribed image points and an exact solution can be obtained. When the number of the prescribed positions is more than five, in general there is no exact solution and we have to seek a best-fit curve in the image space for an approximate solution. The normal curve fitting method (Ravani and Roth, 1983) is used to formulate an error function to represent the summation of the square of the normal distances between prescribed points and the fitted curve in the image space. The solution is found by minimizing the error function using nonlinear least square algorithms (Figure 4-1).

4.2 Curve Fitting in the Planar Quaternion Image Space

A general planar four-bar motion can be mapped onto a curve in the image space. The quadrics in eqs.(3-11 and 3-17)

$$X^T[Q_1(\boldsymbol{r})]X = 0 \text{ , and } X^T[Q_2(\boldsymbol{r})]X = 0 \tag{4-1}$$

are functions of the design parameter vector $\boldsymbol{r} = [r_1, r_2, \ldots r_m]^T$, where m is the total number

of design parameters and $\boldsymbol{X} = [X_1, X_2, X_3, X_4]^T$ is the vector of image space coordinates. The synthesis problem is to determine a curve which passes through, or as close as possible to, a set of n prescribed points $\boldsymbol{X}_d$. We now rewrite eq.(4-1) as

$$Q_1(\boldsymbol{X}, \boldsymbol{r}) = 0$$
$$Q_2(\boldsymbol{X}, \boldsymbol{r}) = 0 \tag{4-2}$$

We also have the constraint equation for the normalized coordinates from the mapping of the planar motion:

$$H = X_3^2 + X_4^2 - 1 = 0 \tag{4-3}$$

Using Taylor series to expand equations (4-2) and (4-3) about a desired point $\boldsymbol{X}_d$ and neglecting higher order terms, we have

$$Q_1 = 0 = Q_1(X_d) + \sum_{i=1}^{4} \left.\frac{\partial Q_1}{\partial X_i}\right|_{X_i = X_{id}} (X_i - X_{id})$$

$$Q_2 = 0 = Q_2(X_d) + \sum_{i=1}^{4} \left.\frac{\partial Q_2}{\partial X_i}\right|_{X_i = X_{id}} (X_i - X_{id}) \tag{4-4}$$

and
$$H = 0 = 2 \sum_{i=3}^{4} [X_{id} (X_i - X_{id})] \tag{4-5}$$

These surfaces are now approximated by the tangent hyperplanes at $\boldsymbol{X}_d$. Combining equations (4-4) and (4-5), we have a set of three equations which can be written in matrix form as

$$J \Delta = V \tag{4-6}$$

where

$$J = \begin{bmatrix} \frac{\partial Q_1}{\partial X_1} & \frac{\partial Q_1}{\partial X_2} & \frac{\partial Q_1}{\partial X_3} & \frac{\partial Q_1}{\partial X_4} \\ \frac{\partial Q_2}{\partial X_1} & \frac{\partial Q_2}{\partial X_2} & \frac{\partial Q_2}{\partial X_3} & \frac{\partial Q_2}{\partial X_4} \\ 0 & 0 & 2X_3 & 2X_4 \end{bmatrix}_{X=X_d} \tag{4-7}$$

$$\Delta = [\Delta X_1 \quad \Delta X_2 \quad \Delta X_3 \quad \Delta X_4]^T \tag{4-8}$$

and

$$v = [\ -Q_1(X_d) \ \ -Q_2(X_d) \ \ 0\ ]^T \tag{4-9}$$

Eq.(4-6) is a system of three equations in four unknowns and thus has an infinite number of solutions, where $\Delta = X - X_d$ is the normal error vector. The distance between a prescribed point X_d and the image curve is defined as the square root of $\Delta^T\Delta$, i.e. $(\Delta X_1^2 + \Delta X_2^2 + \Delta X_3^2 + \Delta X_4^2)^{1/2}$. To determine the minimum distance between each point of X_d and the fitting curve, we can use the minimum-norm pseudo-inverse of J. The error term Δ is obtained by premultiplication of eq.(4-6) by J^T

$$J^T J\, \Delta = J^T V \tag{4-10}$$

and premultiplication by the factor $(J^T J)^{-1}$

$$(J^T J)^{-1} J^T J\, \Delta = (J^T J)^{-1} J^T V \tag{4-11}$$

then

$$\Delta = (J^T J)^{-1} J^T V \tag{4-12}$$

The approximation error function at each point X_d becomes

$$e^2 = \Delta^T \Delta \tag{4-13}$$

The image curve that best fits through all prescribed points is found by minimizing the sum of the normal distance errors at each of the n prescribed image points. The total error is defined as the sum of the squares of each local error function.

$$E = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \Delta^T \Delta \tag{4-14}$$

It should be noted that the errors measured are in the image space and not in the physical space. The design problem becomes the search for a set of design parameters which will minimize eq.(4-14).

4.3 Curve Fitting in the Quaternion Image Space

For spherical four-bar mechanisms, we have two constraint manifold equations similar to eq.(4-1) and the constraint equation for the normalized coordinate of a spherical motion in the image space:

$$H = X_1^2 + X_2^2 + X_3^2 + X_4^2 - 1 = 0 \tag{4-18}$$

We follow the same procedures outlined in the last section to formulate the error function for spherical motions except that the third equation is replaced by eq.(4-18). The minimum norm solution Δ can be obtained from eq.(4-12) and the sum of normal distance error from eq.(4-14).

4.4 Curve Fitting in the Dual Quaternion Image Space

For spatial mechanisms, the curve fitting technique is applied to the dual quaternion image space. A general surface $\hat{Q}$ in dual image space $\hat{I}$ is the locus of the points satisfying an equation of the form

$$\hat{Q}(\hat{X}_1, \hat{X}_2, \hat{X}_3, \hat{X}_4) = 0 \tag{4-19}$$

The above equation can be expanded as

$$Q(X_1, X_2, X_3, X_4) + \varepsilon Q^0(X_1, X_2, X_3, X_4, X_1^0, X_2^0, X_3^0, X_4^0) = 0 \tag{4-20}$$

The primary component and dual component both vanish. Therefore, we have

$$\begin{cases} Q(X) = 0 \\ Q^0(\hat{X}) = 0 \end{cases} \tag{4-21}$$

where $X = (X_1, X_2, X_3, X_4)$ and $\hat{X} = (X_1, X_2, X_3, X_4, X_1^0, X_2^0, X_3^0, X_4^0)$.

A general curve $\hat{k}$ in the dual image space is the locus of points satisfying two dual equations, which imply four real equations:

$$\begin{cases} Q_1(X) = 0 \\ Q_1^0(\hat{X}) = 0 \\ Q_2(X) = 0 \\ Q_2^0(\hat{X}) = 0 \end{cases} \tag{4-22}$$

Expanding the approximating curve $\hat{k}$ in Taylor series about the desired points X_d, or $\hat{X}_d$, and neglecting the higher order terms results in:

$$Q_1 = 0 = Q_1(X_d) + \sum_{i=1}^{4} \left.\frac{\partial Q_1}{\partial X_i}\right|_{X_i = X_{id}} (X_i - X_{id}) \tag{4-23}$$

$$Q_1^0 = 0 = Q_1^0(\hat{X}_d) + \sum_{i=1}^{4} \left[\left.\frac{\partial Q_1^0}{\partial X_i}\right|_{X_i = X_{id}} (X_i - X_{id}) + \left.\frac{\partial Q_1^0}{\partial X_i^0}\right|_{X_i^0 = X_{id}^0} (X_i^0 - X_{id}^0) \right] \tag{4-24}$$

$$Q_2 = 0 = Q_2(X_d) \sum_{i=1}^{4} \left.\frac{\partial Q_2}{\partial X_i}\right|_{X_i = X_{id}} (X_i - X_{id}) \tag{4-25}$$

$$Q_2^0 = 0 = Q_2^0(\hat{X}_d) + \sum_{i=1}^{4} \left[\left.\frac{\partial Q_2^0}{\partial X_i}\right|_{X_i = X_{id}} (X_i - X_{id}) + \left.\frac{\partial Q_2^0}{\partial X_i^0}\right|_{X_i^0 = X_{id}^0} (X_i^0 - X_{id}^0) \right] \tag{4-26}$$

X_d and $\hat{X}_d$ should satisfy two other conditions which are the fundamental relation of the dual quaternion and the condition for the normalized Euler parameters:

$$X_1 X_1^0 + X_2 X_2^0 + X_3 X_3^0 + X_4 X_4^0 = 0 \tag{4-27}$$

$$X_1^2 + X_2^2 + X_3^2 + X_4^2 = 1 \tag{4-28}$$

Expanding eqs.(4-27 and 4-28) in Taylor series about the desired points X_d, or $\hat{X}_d$, and neglecting the higher order terms results in:

$$\sum_{i=1}^{4} \left[X_i^0 (X_i - X_{di}) + X_i (X_i^0 - X_{id}^0) \right] = 0 \tag{4-29}$$

$$2\sum_{i=1}^{4}\left[X_{id}(X_i - X_{id})\right] = 0 \tag{4-30}$$

Combining eqs.(4-23 to 4-28), we have a set of six equations which can be written as a matrix form:

$$J\Delta = V \tag{4-31}$$

where

$$J=\begin{bmatrix} \tilde{Q}_{11} & \tilde{Q}_{12} & \tilde{Q}_{13} & \tilde{Q}_{14} & 0 & 0 & 0 & 0 \\ \tilde{Q}^0_{11} & \tilde{Q}^0_{12} & \tilde{Q}^0_{13} & \tilde{Q}^0_{14} & \tilde{Q}^0_{11^0} & \tilde{Q}^0_{12^0} & \tilde{Q}^0_{13^0} & \tilde{Q}^0_{14^0} \\ \tilde{Q}_{21} & \tilde{Q}_{22} & \tilde{Q}_{23} & \tilde{Q}_{24} & 0 & 0 & 0 & 0 \\ \tilde{Q}^0_{21} & \tilde{Q}^0_{22} & \tilde{Q}^0_{23} & \tilde{Q}^0_{24} & \tilde{Q}^0_{21^0} & \tilde{Q}^0_{22^0} & \tilde{Q}^0_{23^0} & \tilde{Q}^0_{24^0} \\ X^0_{1d} & X^0_{2d} & X^0_{3d} & X^0_{4d} & X_{1d} & X_{2d} & X_{3d} & X_{4d} \\ 2X_{1d} & 2X_{2d} & 2X_{3d} & 2X_{4d} & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4-32}$$

$$\Delta = \begin{bmatrix} \Delta X_1 & \Delta X_2 & \Delta X_3 & \Delta X_4 & \Delta X^0_1 & \Delta X^0_2 & \Delta X^0_3 & \Delta X^0_4 \end{bmatrix}^T \tag{4-33}$$

and

$$V = \begin{bmatrix} -\tilde{Q}_1 & -\tilde{Q}^0_1 & -\tilde{Q}_2 & -\tilde{Q}^0_2 & 0 & 0 \end{bmatrix}^T \tag{4-34}$$

with

$$\tilde{Q}_1 = Q_1(X_d), \tilde{Q}^0_1 = Q^0_1(X_d), \tilde{Q}_2 = Q_2(X_d), \tilde{Q}^0_2 = Q^0_2(X_d) \tag{4-35}$$

and

$$\left\{ \begin{array}{l} \Delta X_i = X_i - X_{id}, \quad \Delta X^0_i = X^0_i - X^0_{id} \\ \tilde{Q}_{1i} = \left.\dfrac{\partial Q_1}{\partial X_i}\right|_{X_i = X_{id}}, \quad \tilde{Q}^0_{1i} = \left.\dfrac{\partial Q^0_1}{\partial X_i}\right|_{X_i = X_{id}}, \quad \tilde{Q}_{1i^0} = \left.\dfrac{\partial Q^0_1}{\partial X^0_i}\right|_{X^0_i = X^0_{id}} \\ \tilde{Q}_{2i} = \left.\dfrac{\partial Q_2}{\partial X_i}\right|_{X_i = X_{id}}, \quad \tilde{Q}^0_{2i} = \left.\dfrac{\partial Q^0_2}{\partial X_i}\right|_{X_i = X_{id}}, \quad \tilde{Q}_{2i^0} = \left.\dfrac{\partial Q^0_2}{\partial X^0_i}\right|_{X^0_i = X^0_{id}} \end{array} \right\} \quad i = 1,2,\ldots,4 \tag{4-36}$$

Eq.(4-31) is a system of six equations in eight unknowns and thus has an infinite number of solutions. To determine the minimum distance between each point of X_d, or $\hat{X}_d$, and

the curve $\hat{k}$, we again need to determine the minimum norm solution of eq.(4-31). In view of equations (4-12), the minimum norm solution Δ can also be obtained as

$$\Delta = (J^T J)^{-1} J^T V \tag{4-37}$$

The approximation error function at each point X_d, or $\hat{X}_d$, becomes

$$e^2 = \Delta^T \Delta$$

The curve fitting of the given points is found by minimizing the sum of the normal distance errors at each of n prescribed image points:

$$E = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \Delta^T \Delta \tag{4-39}$$

CHAPTER 5

GLOBAL MINIMIZATION METHODS

The method presented in this study involves a search for the minimization of the error function which consists of a total least square problem. Three algorithms were explored for the purpose of comparison: one is the well known Levenberg-Marquart algorithm, and the other two are newly developed artificial intelligence searching algorithms: the genetic algorithms and the simulated annealing methods.

5.1 Levenberg-Marquardt Algorithm

The Levenberg-Marquardt (L-M) Algorithm (Dennis and Schnabel, 1983) was developed to solve nonlinear least squares problems. This algorithm is a modification of the Gauss-Newton method. The problem is stated as follows:

$$\min_{x \in R^n} \frac{1}{2} F(x)^T F(x) = \frac{1}{2} \sum_{i=1}^{m} f_i(x)^2$$

where m $\geq$ n, F: $\boldsymbol{R}^n \rightarrow \boldsymbol{R}^m$, and $f_i(x)$ is the i-th component function of $F(x)$. From a current point x_c, the algorithm uses the trust region approach:

$$\min_{x \in R^n} \| F(x_c) + J(x_c)(x_n - x_c) \|_2$$

$$\text{subject to} \qquad \| x_n - x_c \|_2 \leq \delta_c.$$

where δ_c is the maximal step length, or the trust radius.

A new point x_n is obtained by:

$$x_n = x_c - (J(x_c)^T J(x_c) + \mu_c I)^{-1} J(x_c)^T F(x_c),$$

where μ_c is a parameter to control the direction toward the steepest-decent direction and the step size. $\mu_c = 0$ if $\delta_c \geq \| (J(x_c)^T J(x_c))^{-1} J(x_c)^T F(x_c) \|_2$ and $\mu_c > 0$ otherwise. $F(x_c)$ and $J(x_c)$ are the function values and the Jacobian evaluated at the current point x_c. This procedure is repeated until the stopping criteria are satisfied. The L-M algorithm

may still be slowly convergent on large residual or very nonlinear problems. However, many implementations of this algorithm, for example, MINPACK by Moré who used a scaled trust region to choose μ_c and δ_c or the IMSL routine ZXSSQ or UNLSF, have proven to be very successful in practice. There are some merits that make the L-M algorithm better than the damped Gauss-Newton method on many problems. One is that the L-M method is well defined even when $J(x_c)$ does not have full column rank. Another is that when the Gauss-Newton step is much too long, the L-M step is close to being in the steepest-descent direction $-J(x_c)^T F(x_c)$. The Fortran program using L-M algorithm for synthesis of planar mechanisms is listed in Appendix C.

5.2 Genetic Algorithms

Genetic algorithms (GA's) are adaptive search and optimization techniques based on the theory of natural selection and natural genetics (Goldberg 1989, Schraudolph and Grefenstette, 1992). As function optimizers, GA's are computational models that emulate biological evolutionary theories to solve optimization problems and are iterative procedures which maintain a population $\boldsymbol{P}$ of n candidate solutions of an objective function f:

$$\boldsymbol{P}(t) = \{\boldsymbol{x}_1(t), \boldsymbol{x}_2(t), \ldots\ldots, \boldsymbol{x}_n(t)\}$$

Each candidate structure $\boldsymbol{x}_i$ in population $\boldsymbol{P}(t)$ at time t is simply a binary string of length l, where l is the length of the coded design parameters analogous to a chromosome in genetics. Each $\boldsymbol{x}_i$ represents a vector of parameters of the function $f(\boldsymbol{x})$, but the semantics associated with the vector is unknown to the GA. During each iteration step, called a *generation*, the performance (or fitness) of each individual of the current population is evaluated according to a fitness function, and, on the basis of that evaluation, a new generation of candidate solutions is formed by some genetic manipulations including: *reproductions*, *crossovers*, and *mutations*. Termination of the search procedure may be

triggered by finding an acceptable approximate solution to $f(x)$, by fixing the total number of evaluations, or some other application-dependent criteria.

A general schematic of this procedure is shown in Figure 5-1 (Filho, Alippi, and Treleaven 1993).

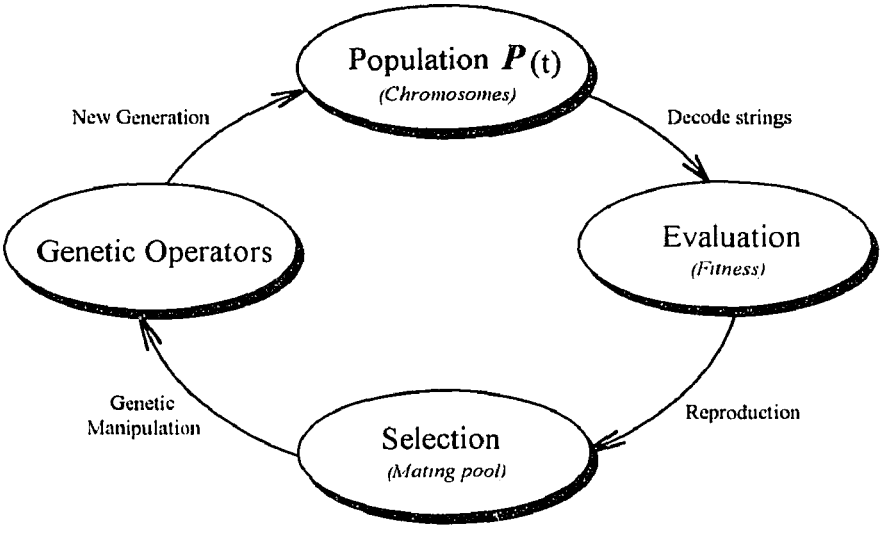


Figure 5-1 A typical GA cycle.

A typical GA contains the following steps:

1. Randomly generate a population of strings,
2. Evaluate each string based on a fitness function,
3. Select some good strings according to the evaluation,
4. Perform genetic manipulations,
5. Create a new population of string.

5.2.1 Reproduction

The first generation $P(0)$ of this process is randomly generated as a starting point for the search process. From there on, the genetic operations, in concert with the fitness function, operate to improve the generation of potential solutions. The first genetic operation is reproduction which is to produce the next generation based on the evaluation of each individual of the previous generation by a selection procedure. Those who have better performance in the parent generation will have a higher probability to be appear in the

child generation. That is, if x_j has twice the average performance of all the structures in $P(t)$, then it is expected to appear twice in population $P(t+1)$. At the end of the selection procedure, population $\mathbf{P}(t+1)$ contains exact duplicates of the selected structures in population $\mathbf{P}(t)$.

5.2.2 Crossover

In order to search other points in the search space, some variation is introduced into the new population by means of idealized genetic operators. The most important operator is called *crossover*. Under the crossover operator, two structures in the new population exchange portions of their binary representation. This can be implemented by choosing a point at random, called the *crossover point*, and exchanging the segments to the right of this point. Figure 5-2 shows the crossover operation between two chromosomes.

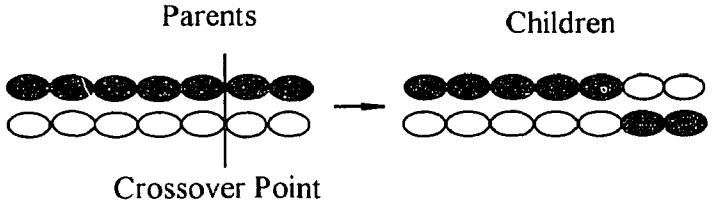


Figure 5-2 Crossover operation between two chromosomes.

5.2.3 Mutation

Mutation is another genetic operator and is implemented by occasionally altering a random bit in a string. Figure 5-3 presents the mutation operator being applied to the sixth element of the string.

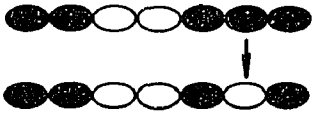


Figure 5-3 Mutation operation of a chromosome.

The description of the GA given in this section presents an overall idea of the steps needed to design a genetic algorithm. However, real implementations have to consider a number of problem-dependent parameters such as the population size, crossover and mutation rate, convergence criteria, etc. GA's are very sensitive to most of these parameters. Methods of setting them up is still an active research topic. In our study, we had difficulties to obtain the optimal solution without reducing the number of design variables. We tried different values of these genetic parameters but no clear trend is found. Using genetic algorithms for optimization problems can be found in Bäck and Schwefel (1993), Leu, wong, and Ji (1993) and many others.

5.3 Simulated Annealing Methods

The adaptive simulated annealing (ASA) algorithm developed by Ingber (1993) was also used in this study. Simulated annealing (SA), motivated by an analogy to the physical process of annealing a material, uses temperature cooling operations to transform an initial poor solution into a high quality global solution. The SA algorithm starts at some high temperature T_o given by the user. Let f be the function to minimize. SA conducts an iterative random search procedure with adaptive moves along the coordinate directions from an arbitrarily selected solution and then evaluates the resulting change in the objective function Δf. If $\Delta f \leq 0$, the new solution is accepted as the starting point for the next move. However, if $\Delta f > 0$, the move is accepted with probability $P(\Delta f) = e^{-\Delta f/T}$ which permits uphill moves under the control of the probabilistic criteria, where T is the temperature which controls the state generation and state acceptance. Therefore, a move toward a state which increases the objective function can be accepted occasionally. This uphill move allows the procedure to escape from local minima and more effectively search the function space to find the global minimum. By gradually lowering the temperature, fewer uphill moves are allowed, and the likelihood that the solution approaches a global optimum increases. Finally, when the temperature reaches zero, only downhill moves are

permitted and the system finds the lowest minimum. The ASA has exponentially decreasing temperature T schema with respect to annealing time t which is faster than the traditional (Boltzmann) annealing where annealing schedule decreases logarithmlically. Recent research (Ingber, 1989, Ingber and Rosen, 1993) has shown that SA is more efficient than a standard genetic algorithm.

CHAPTER 6

NUMERICAL EXAMPLES

Numerical examples are presented to complete the whole idea of this approach. We still start with planar mechanisms which are displayed numerically and graphically. For spherical mechanisms only numerical data are shown.

6.1 Construction of Planar Mechanisms

After obtaining the numerical results from the optimization routine, we can then construct the mechanism. The set of design parameters for planar mechanisms are $X = [x_1, y_1, a_1, b_1, x_2, y_2, a_2, b_2, h, \eta]$. The parameters (x_1, y_1) and (x_2, y_2) are coordinates of the two fixed pivots and the parameters a_1 and a_2 are the link lengths of the input and the output cranks which are also the radii for drawing the crank circles. The parameters used for constructing the coupler are lengths b_1, b_2, and h. The orientation of the tracer frame with respect to the fixed frame is denoted by the angle η.

There are no constraints associated with these parameters except that a_1 and a_2 have to be positive in order to draw the crank circles. The configurations of the coupler for positive or negative b_1, b_2, h, and η are shown in Figure 6-1, 6-2, 6-3, and 6-4, respectively.

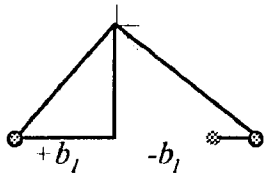


Figure 6-1 The configuration of the coupler for a positive or a negative b_1.

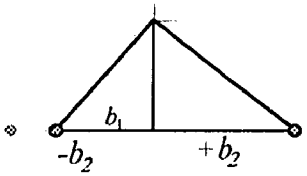


Figure 6-2 The configuration of the coupler for a positive or a negative b_2.

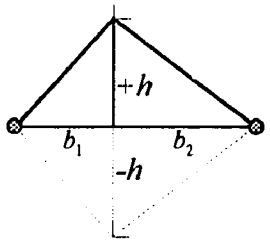


Figure 6-3 The configuration of the coupler for a positive or a negative h.

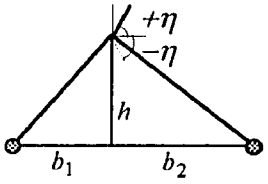


Figure 6-4 The configuration of the tracer frame for a positive or a negative η.

To verify the synthesis of the mechanism, we simply draw the tracer frame and the coupler at each and every prescribed position and orientation. If a valid solution exists, the moving pivots should be located on, or as close as possible to, the path of the crank circles. A program written in Quickbasic is also developed to animate the motion of the planar four-bar mechanism for verification.

6.2 Planar Mechanisms Examples

For motion synthesis problems, if the number of positions and orientations is less than five, graphical methods, analytical methods or software packages work well. Numerical methods are suitable for problems with five or more positions. Five, six, and ten positions are presented in the following examples using Levenberg-Marquart algorithm to illustrate our approach.

Example 1. For a five-position synthesis problem, the prescribed orientations and positions and their corresponding image point coordinates are listed in Table 6-1.

As we mentioned in last chapter, the results of the global algorithms are greatly affected by the initial guesses. In the examples we used the Lincages-4© software to obtain the fixed pivots, (x_1, y_1) and (x_2, y_2), and crank lengths, a_1 and a_2, for any

combination of four positions out of the given five positions as our initial guesses for these parameters. There are a total of five different initial guesses as shown in Table 6-2. Since there are an infinite number of solutions for the four positions synthesis, we still have free choices. The parameters b_1, b_2, h, and η associated with the dimensions of the coupler are random guesses. The solutions for five positions synthesis are listed in Table 6-3. Two solutions, case 1 and 2, are illustrated in Figure 6-1. Other solutions are shown in Appendix-A.

Table 6-1 The prescribed situations (θ_i, a_i, b_i) and their corresponding image point coordinates X_i, i = 1, 2,..4, for example 1.

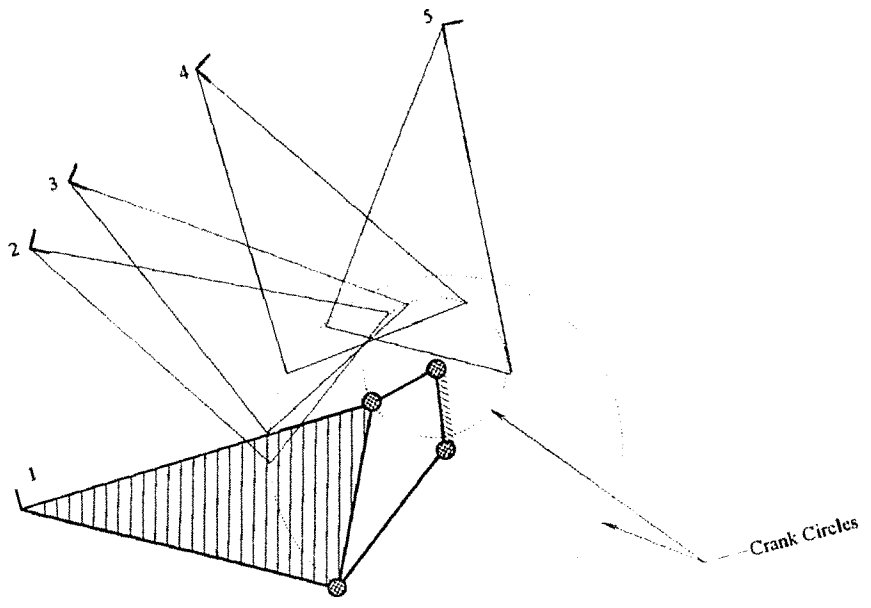
Position	θ (deg)	a (cm)	b (cm)	X_1	X_2	X_3	X_4
1	0.0	-19.21	-1.68	-9.605	-0.840	0.000	1.000
2	-27.3	-15.34	11.38	-8.796	3.719	-0.236	0.972
3	-37.0	-12.58	14.28	-8.231	4.775	-0.317	0.948
4	-58.1	-4.83	18.28	-6.549	6.817	-0.486	0.874
5	-94.9	7.88	17.09	-3.631	8.681	-0.737	0.676

Table 6-2 Initial guesses of the design parameters for the five-position example.

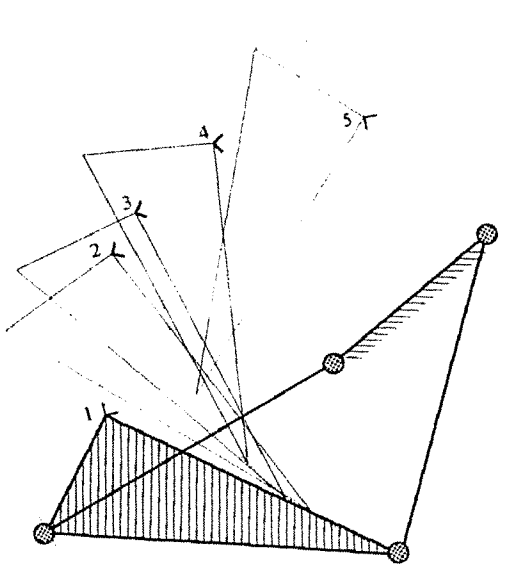
Case	Pos.	x_1	y_1	a_1	b_1	x_2	y_2	a_2	b_2	h	η
1	1234	4.67	-7.0	14.04	10.0	1.78	-4.68	2.59	15.0	24.0	0.0
2	1235	5.36	-4.72	33.21	10.0	19.12	7.68	31.69	10.0	10.0	0.0
3	1245	0.49	-3.06	66.69	30.0	2.62	-0.02	4.38	30.0	20.0	0.0
4	1345	1.69	-4.31	13.54	10.0	3.16	-0.73	3.21	10.0	10.0	0.0
5	2345	3.18	-2.78	1.91	30.0	0.21	-2.84	2.56	30.0	10.0	-10.0

Table 6-3 Solutions for the five-position example for different initial guesses.

	Design parameters										
Cases	x_1	y_1	a_1	b_1	x_2	y_2	a_2	b_2	h	η	error
1	0.32	-2.72	14.04	0.85	3.51	1.69	7.93	13.19	8.73	-52.58	1.50x10-3
2	5.11	-5.51	39.63	13.41	28.83	16.77	45.98	18.20	22.46	-11.55	6.94x10-3
3	0.72	-0.80	7.30	-12.88	-4.74	-1.53	12.73	27.10	1.14	1.41	5.50x10-3
4	3.70	-5.06	45.34	31.05	1.49	-4.15	26.07	-12.05	6.25	-50.57	8.98x10-4
5	0.47	-3.21	18.10	3.59	2.78	1.05	7.06	12.99	8.01	-43.70	6.04x10-4



Case 1



Case 2

Figure 6-5 Two solutions for the five-position synthesis example.

Example 2. Another situation is added to those of example 1 to consider the case of six-position synthesis (Table 6-4). It is impossible to synthesize a six-position problem using analytical methods. We again used the Lincages-4© software to obtain the fixed pivots, (x_1, y_1) and (x_2, y_2), and the crank lengths, a_1 and a_2, for any combination of four positions out of the six positions as our initial guesses for these parameters. There are a total of fifteen different combinations as shown in Table 6-5.

Table 6-4 The prescribed situations and their corresponding image point coordinates for the six-position problem.

Position	θ (deg)	a (cm)	b (cm)	X_1	X_2	X_3	X_4
1	0.0	-19.21	-1.68	-9.605	-0.840	0.000	1.000
2	-27.3	-15.34	11.38	-8.796	3.719	-0.236	0.972
3	-37.0	-12.58	14.28	-8.231	4.775	-0.317	0.948
4	-58.1	-4.83	18.28	-6.549	6.817	-0.486	0.874
5	-94.9	7.88	17.09	-3.631	8.681	-0.737	0.676
6	200.0	8.78	7.91	1.214	-10.116	0.961	-0.276

Table 6-5 Initial guesses of design parameters for the six-position example.

Case	Pos.	x_1	y_1	a_1	b_1	x_2	y_2	a_2	b_2	h	η
1	1234	4.67	-7.0	14.04	10.0	1.78	-4.68	2.59	15.0	24.0	0.0
2	1235	5.36	-4.72	33.21	10.0	19.12	7.68	31.69	10.0	10.0	0.0
3	1236	0.22	7.46	30.19	30.0	-4.36	-2.14	17.89	30.0	40.0	-50.0
4	1245	0.49	-3.06	66.69	30.0	2.62	-0.02	4.38	30.0	20.0	0.0
5	1246	0.85	-2.86	14.25	10.0	2.05	0.70	6.52	10.0	10.0	0.0
6	1256	2.15	-4.41	30.59	10.0	20.85	19.38	41.87	10.0	10.0	0.0
7	1345	1.69	-4.31	13.54	10.0	3.16	-0.73	3.21	10.0	10.0	0.0
8	1346	1.02	-3.91	28.06	5.0	-4.35	0.36	33.67	5.0	5.0	-10.0
9	1356	1.20	-3.92	27.37	10.0	-3.83	4.55	23.84	10.0	10.0	0.0
10	1456	1.63	-4.18	30.63	10.0	-3.27	4.99	17.62	10.0	10.0	0.0
11	2345	3.18	-2.78	1.91	30.0	0.21	-2.84	2.56	30.0	10.0	-10.0
12	2346	3.28	-3.08	12.41	10.0	-0.76	-1.35	12.96	10.0	10.0	0.0
13	2356	1.91	0.05	6.75	10.0	-0.34	-1.43	11.76	10.0	10.0	0.0
14	2456	4.45	0.08	11.21	10.0	-4.44	3.74	8.87	10.0	10.0	-10.0
15	3456	4.67	0.33	11.72	10.0	-4.59	4.03	8.16	10.0	10.0	0.0

Table 6-6 Solutions for the six-position example with different initial guesses (EPS= 10^{-6}).

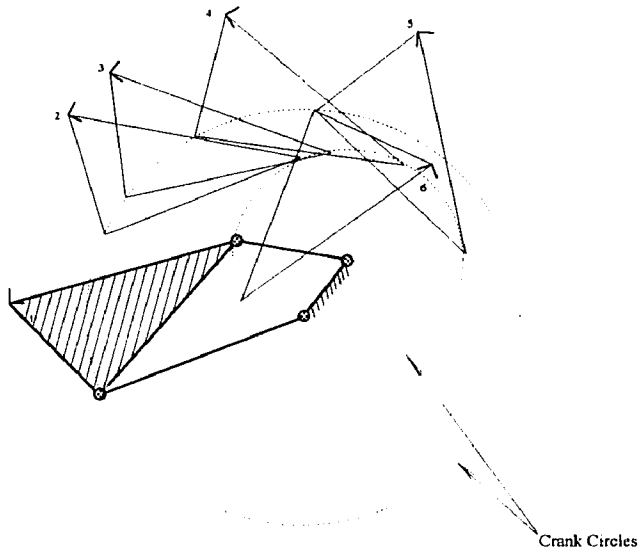
	Design parameters										
	x_1	y_1	a_1	b_1	x_2	y_2	a_2	b_2	h	η	error
1	0.322	-2.724	14.038	0.845	3.510	1.690	7.932	13.185	8.732	-52.582	1.50x10-3
2	5.110	-5.506	39.635	13.405	28.826	16.766	45.977	18.196	22.459	-11.546	6.94x10-3
3	0.720	-0.796	7.303	-12.879	-4.736	-1.534	12.732	27.099	1.136	1.413	5.03x10-3
4	3.347	-4.662	53.399	38.850	2.233	0.700	6.652	12.064	8.893	-44.548	4.64x10-4
5	0.466	-3.206	18.101	3.592	2.783	1.045	7.061	12.991	8.013	-43.799	6.04x10-4
6	2.171	-4.413	34.076	10.676	23.892	22.335	43.507	22.280	17.884	-6.414	1.17x10-3
7	0.508	-3.139	18.808	4.161	2.370	0.801	6.728	12.678	8.088	-41.640	4.01x10-4
8	3.364	1.402	7.754	13.179	0.223	-2.576	14.045	0.738	-8.682	129.18	9.07x10-4
9	0.907	-3.628	21.288	6.381	2.501	0.894	6.819	12.828	7.981	-41.930	3.79x10-4
10	1.432	-4.017	26.859	11.808	2.251	0.723	6.655	12.596	8.066	-40.888	2.91x10-4
11	-0.511	-2.622	20.886	7.935	6.102	-1.282	19.325	27.953	5.592	-51.228	8.51x10-3
12	0.188	-2.895	15.979	1.823	2.934	1.100	7.188	12.938	8.252	-45.619	1.91x10-3
13	0.800	-3.533	22.020	7.158	2.414	0.808	6.781	12.818	7.907	-41.136	2.49x10-4
14	0.605	-0.893	7.298	-12.977	-4.516	-0.584	14.555	28.939	1.009	1.607	4.66x10-3
15	0.668	-1.016	7.32	-12.85	-4.36	-1.25	15.92	30.63	1.89	-1.71	5.93x10-3

Table 6-7 Solutions for the six-position example with different initial guesses (EPS= 10^{-7}).

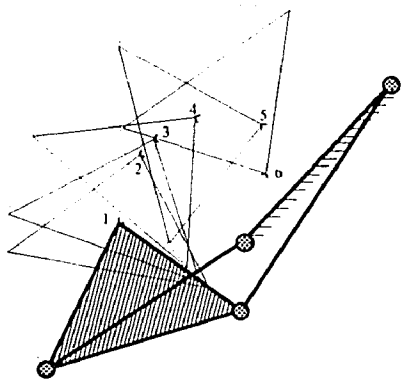
	Design parameters										
Case	x_1	y_1	a_1	b_1	x_2	y_2	a_2	b_2	h	η	error
1	0.234	-2.603	14.373	0.864	3.240	1.345	7.636	13.137	8.595	-49.51	8.56x10-4
2	4.348	-5.286	44.671	21.089	32.120	24.497	51.749	17.462	22.645	-17.09	2.20x10-3
3	0.720	-0.796	7.303	-12.879	-4.736	-1.534	12.732	27.099	1.136	1.413	5.03x10-3
4	3.347	-4.662	53.399	38.850	2.233	0.700	6.652	12.064	8.893	-44.55	4.64x10-4
5	0.358	-2.953	16.456	2.229	2.855	1.111	7.157	12.989	8.214	-45.21	5.28x10-4
6	0.213	-2.743	15.481	-7.347	17.239	19.514	38.740	30.446	4.132	24.66	1.15x10-3
7	0.190	-2.685	14.466	0.985	3.282	1.216	7.616	13.093	8.623	-49.31	1.19x10-3
8	0.152	-2.438	12.941	3.056	8.737	0.682	47.170	53.710	8.471	-73.02	2.10x10-3
9	0.332	-2.758	14.576	1.027	3.245	1.438	7.584	13.080	8.602	-49.97	8.78x10-4
10	1.432	-4.017	26.859	11.808	2.251	0.723	6.655	12.596	8.066	-40.888	2.91x10-4
11	-0.511	-2.622	20.886	7.935	6.102	-1.282	19.325	27.953	5.592	-51.23	8.51x10-3
12	0.306	-2.726	14.508	0.966	3.293	1.447	7.661	13.093	8.603	-50.05	8.82x10-4
13	0.175	-2.626	14.298	0.901	3.423	1.307	7.801	13.224	8.611	-50.10	8.71x10-4
14	0.667	-0.927	7.281	-12.94	-4.517	-1.164	14.513	29.069	1.335	0.30	4.43x10-3
15	-0.030	-2.733	17.511	-4.835	10.341	17.451	32.831	30.312	7.013	14.84	1.97x10-3

Solutions for the six-position problem are listed in Table 6-6 and Table 6-7 by using different values of EPS in the stop criteria. EPS is the second stop criterion in the IMSL routine. The program convergence condition is satisfied if on two successive iterations, the residual sum of square estimates have relative difference less than or equal to EPS. In Table 6-6 EPS = 10^{-6} is used as the stop criterion, while EPS = 10^{-7} is used in Table 6-7.

Four graphical solutions are shown in Figures 6-6 and Figure 6-7 for cases 1, 2, 3, and 4. Other graphical solutions are shown in Appendix-B.

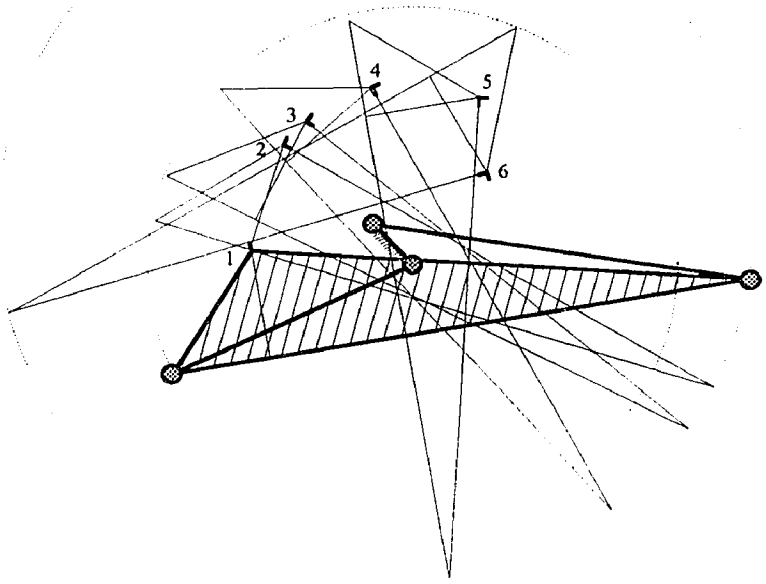


Case 1

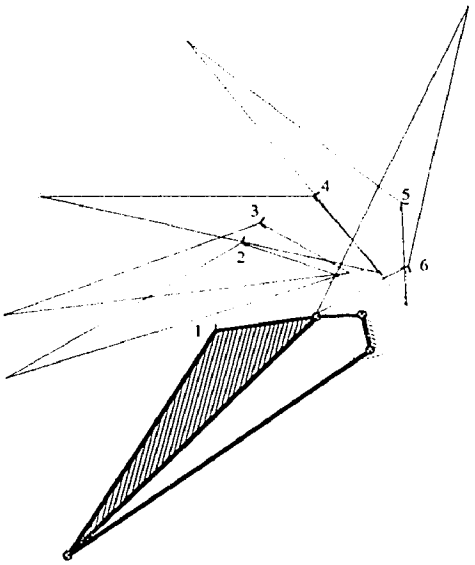


Case 2

Figure 6-6 Solutions for the six-position synthesis problem.



Case 3



Case 4

Figure 6-7 Solutions for six position synthesis problem.

Example 3 Five more situations are added to the five situations of example 1 for a ten-position synthesis problem as listed in Table 6-8. Five runs with different initial guesses are tabulated in Table 6-9. It has been observed that some solutions resulted in essentially the same values as other solutions but with the order of the parameters varied. After rearrangement, they all converged approximately to the same set of design parameters. The average values of the design parameters and their standard deviations are listed in Table 6-10. The graphical solution for the average values of design parameters is shown in Figure 6-8. A typical convergence diagram for the ten-position problem is shown in Figure 6-9. The convergence curves for five and ten-position problems, using the ASA algorithm, are shown in Figure 6-10. The design parameters found using the ASA algorithm is listed in Table 6-11.

Table 6-8 The situations and their image point coordinates for ten positions.

Position	θ (deg)	a (cm)	b (cm)	X_1	X_2	X_3	X_4
1	0.0	-19.21	-1.68	-9.605	-0.840	0.000	1.000
2	-27.3	-15.34	11.38	-8.796	3.719	-0.236	0.972
3	-37.0	-12.58	14.28	-8.231	4.775	-0.317	0.948
4	-58.1	-4.83	18.28	-6.549	6.817	-0.486	0.874
5	-94.9	7.88	17.09	-3.631	8.681	-0.737	0.676
6	212.0	18.78	7.91	1.214	-10.116	0.961	-0.276
7	171.8	21.80	-0.06	0.749	-10.874	0.997	0.071
8	92.0	12.55	-8.07	1.456	-7.317	0.719	0.695
9	20.0	-3.47	-12.76	-2.817	-5.982	0.174	0.985
10	6.0	-18.33	-5.79	-9.304	-2.411	0.052	0.999

Table 6-9 Solutions for the ten-position example with different initial guesses.

Case	Design parameters										error
	x_1	y_1	a_1	b_1	x_2	y_2	a_2	b_2	h	η	
1234	10.460	-2.476	20.702	19.436	1.268	0.777	9.103	-0.656	11.545	-103.91	8.67x10-4
1235	10.459	-2.452	20.578	19.384	1.247	0.794	9.093	-0.694	11.535	-104.27	9.86x10-4
1245	1.255	0.817	9.112	0.776	10.449	-2.468	20.216	-19.106	11.523	-104.71	1.85x10-3
1345	10.494	-2.468	20.357	19.223	1.237	0.812	9.111	-0.748	11.530	-104.63	1.62x10-3
2345	1.267	0.758	9.107	-0.843	10.565	-2.439	20.574	19.516	-11.524	75.356	2.36x10-3
The solutions for case 1245 and 2345 can be rearranged as:											
1245	10.449	-2.468	20.216	19.106	1.255	0.817	9.112	-0.776	11.523	-104.71	1.85x10-3
2345	10.565	-2.439	20.574	19.516	1.267	0.758	9.107	-0.843	11.524	-104.64	2.36x10-3

Table 6-10 The average values $\bar{X}$ of the design parameters and their standard deviations σ.

	The average design parameters										
	x_1	y_1	a_1	b_1	x_2	y_2	a_2	b_2	h	η	error
$\bar{X}$	10.485	-2.461	20.485	19.333	1.255	0.791	9.105	0.743	11.531	-104.43	1.54x10-3
σ	0.0475	0.0149	0.1953	0.1659	0.0133	0.0246	0.0077	0.0727	0.0090	0.3410	0.00062

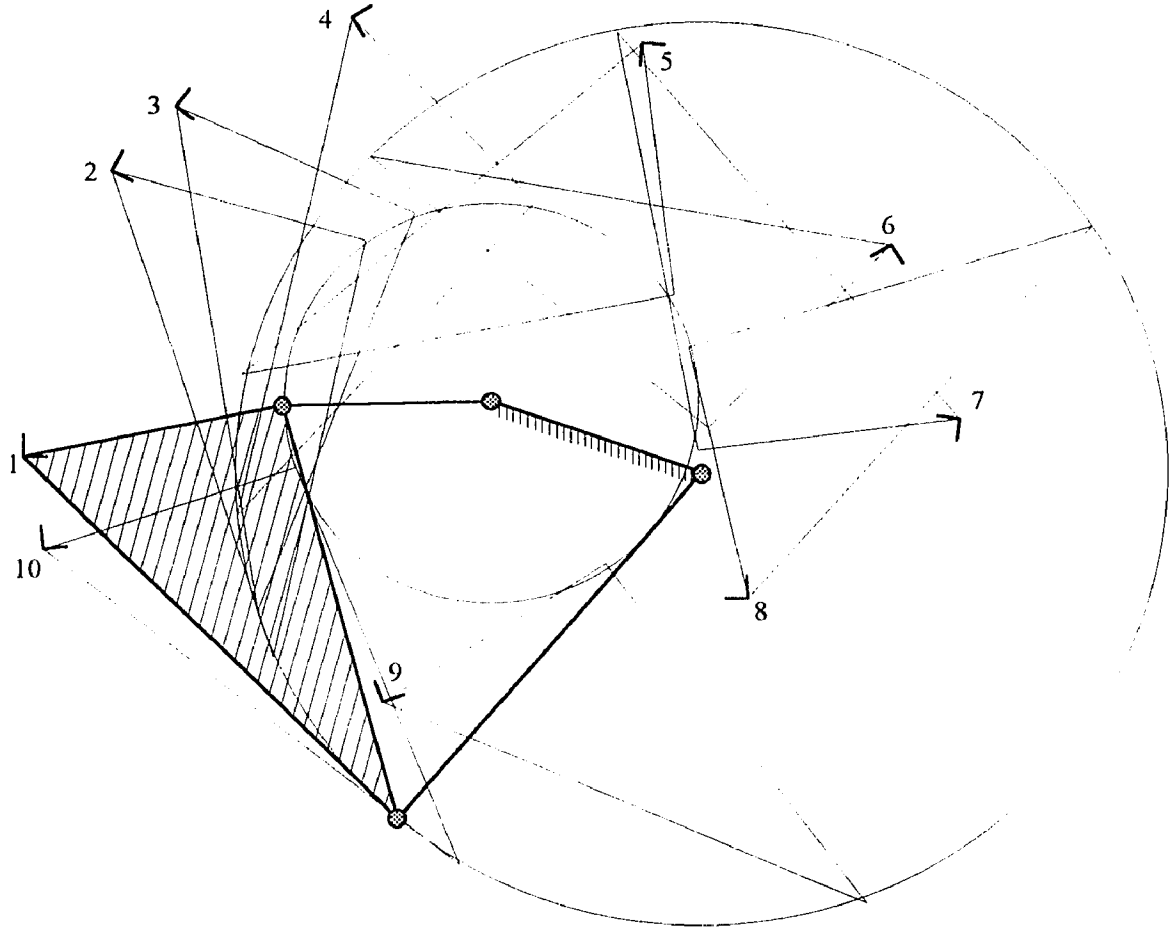


Figure 6-8 The mechanism found to pass closely to the ten prescribed situations.

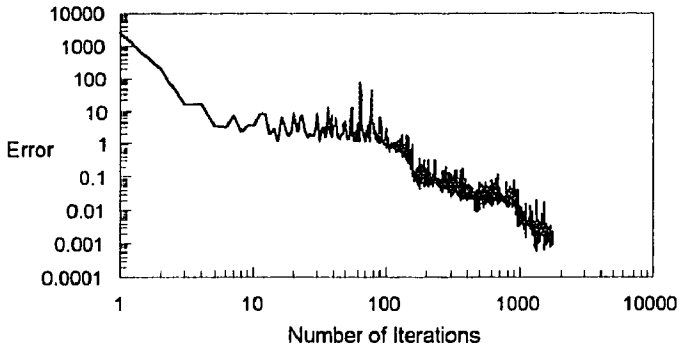


Figure 6-9 A typical convergence diagram for the ten-position example.

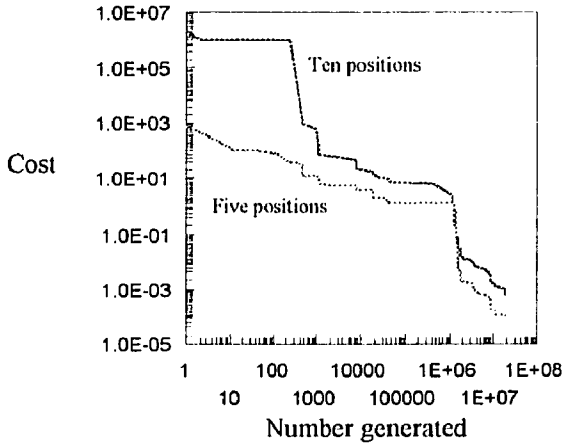


Figure 6-10 The convergence curves using the ASA algorithm.

Table 6-11 Optimal design parameters for five and ten positions.

	Design parameters										
	x_1	y_1	a_1	b_1	x_2	y_2	a_2	b_2	h	η	error
5*	10.01	-2.05	24.06	22.11	1.34	0.80	9.04	-0.80	11.73	-98.89	9.8x10-5
10*	10.47	-2.42	20.64	19.44	1.26	0.79	9.08	-0.69	11.55	-104.1	6.1x10-4

* Using the ASA algorithm.

6.3 Numerical Examples for Spherical Mechanisms

Numerical examples for five and ten positions synthesis of a spherical mechanism as discussed in chapter 3.2 are listed in Table 6-12 and Table 6-13. These examples are given in Bodduluri and McCarthy (1992). The optimal design parameters for the spherical mechanism is listed in Table 6-14.

Table 6-12 The prescribed rotation axes and angles and their corresponding image points for the five-position synthesis example.

	Rotation axis			Angle	Image Point			
No.	u_x	u_y	u_z	(deg)	x_1	x_2	x_3	x_4
1	0.000	0.000	0.000	0.00	0.000	0.000	0.000	1.000
2	0.124	-0.978	-0.164	25.66	0.028	0..217	-0.036	0.975
3	-0.018	-0.999	0.025	71.73	-0.011	-0.586	0.015	0.081
4	-0.762	-0.648	-0.001	80.20	-0.491	-0.417	-0.005	0.765
5	-1.000	0.000	0.000	89.10	-0.702	0.000	0.000	0.713

Table 6-13 The prescribed rotation axes and angles and their corresponding image points for the ten-position example.

	Rotation axis			Angle	Image Point			
No.	u_x	u_y	u_z	(deg)	x_1	x_2	x_3	x_4
1	0.000	0.000	0.000	0.00	0.000	0.000	0.000	1.000
2	0.143	-0.966	-0.215	22.38	0.028	-0.187	-0.042	0.981
3	0.060	-0.995	-0.077	48.34	0.025	-0.41	-0.031	0.912
4	-0.018	-0.999	0.025	71.73	-0.011	-0.586	0.015	0.810
5	-0.484	-0.873	0.051	82.59	-0.320	-0.576	0.033	0.751
6	-0.762	0-0.648	-0.001	80.20	-0.491	-0.417	-0.005	0.765
7	-1.000	0.000	0.000	89.10	-0.702	0.000	0.000	0.713
8	-0.767	0.460	0.448	63.35	-0.403	0.242	0.235	0.851
9	-0.578	0.623	0.528	43.31	-0.213	0.230	0.195	0.929
10	-0.369	0.801	0.470	18.95	-0.061	0.132	0.077	0.986

Table 6-14 The optimal design parameters for spherical four-bar mechanisms.

	Design parameters										
	ψ_1	α_1	β_1	γ_1	ψ_2	α_2	β_2	γ_2	σ	η	error
5*	33.11	35.11	42.05	105.03	166.06	93.78	62.63	126.07	159.84	154.67	8.56x10-9
10*	34.41	34.10	41.87	-106.46	157.28	93.72	72.08	-124.85	-157.20	-28.41	3.75x10-7

* Using the ASA algorithm.

CHAPTER 7

CONCLUSIONS AND DISCUSSIONS

This study furthers the development of curve-fitting in the image space by using constraint manifold equations as a technique for the motion synthesis of mechanisms. There are some basic advantages in using constraint manifold equations for solving mechanism synthesis problems. First, they can be derived easily using quaternion algebra. Second, they are functions of the link and joint variables of the mechanism, which are the design parameters for the motion synthesis problem. Finally, the constraint manifolds are actually the structure equations resulting from the relationship of the tracer frame to the reference frame which represents the geometry of the mechanism explicitly.

Integration of kinematic mappings with the theories of quaternion algebra is one of the contributions of this study to the field of kinematic synthesis. The general idea is to convert the prescribed situations in real space to points in an image space. By using quaternion algebra we then curve fit a system of equations to the points in the image space. The parameters of these equations are the dimensions required to construct the mechanism. Dual quaternions that represent both rotation and translation for a screw motion are applied to synthesize general spatial mechanisms. For spherical mechanisms that involve only rotational motion, quaternions are used to identify the motion. Planar quaternions are used for planar mechanisms.

As in all synthesis techniques, the mechanisms developed by the method described in this study may exhibit branch defects, meaning that the linkage may not be able to move from one prescribed position to the next although it can be disconnected and reassembled in each prescribed position. Besides branch defects, there may be order defects: the synthesized linkage may pass through the prescribed situations but not in the prescribed order. Non-branch constraint and linkage type constraints, for example, Grashof and non-

Grashof constraints, can be formulated in the unconstrained optimization problem by using penalty functions (Rao, 1984). This technique has been implemented by Bodduluri and McCarthy (1992). Papers dealing with branch problems include: Waldron, (1977), Reinholtz, Sandor, and Duffy, (1986) and others.

Using numerical minimization algorithms the method developed can be applied for any number of prescribed positions and orientations, while analytical or graphical methods are generally limited up to five prescribed situations.

7.1 Discussions of Numerical Results

It is well known that when the total number of prescribed situations is five or less, an image curve can be found to pass through these prescribed image points and an exact solution can be obtained. When the number of the prescribed situations is more than five, in general there is no exact solution and we have to seek a best-fit curve in the image space for an approximate solution.

The numerical solutions for five, six, and ten positions have been illustrated in chapter 6. The resulting mechanisms approximate the prescribed situations by using nonlinear least square algorithms to minimize the normal distance errors between all the desired image points and image curve of the tracer frame. As a result, depending upon the allowable error in the minimization algorithm, an infinite number of solutions could be found. An obvious advantage for having more than one solution, even it is not exact, is to give the designer flexibility of choices. The solutions to which the algorithm converges depend upon the stopping criteria adapted and the initial guesses. The method to select the initial guesses has been explained in detail in chapter 6. As far as the stopping criteria are concerned, there are three stopping criteria in the ZXSSQ routine provided in the IMSL package which was used. The first convergence criterion is NSIG. The convergence condition is satisfied, if on two successive iterations the parameters estimates agree, component by component, to NSIG. The second convergence criterion is EPS.

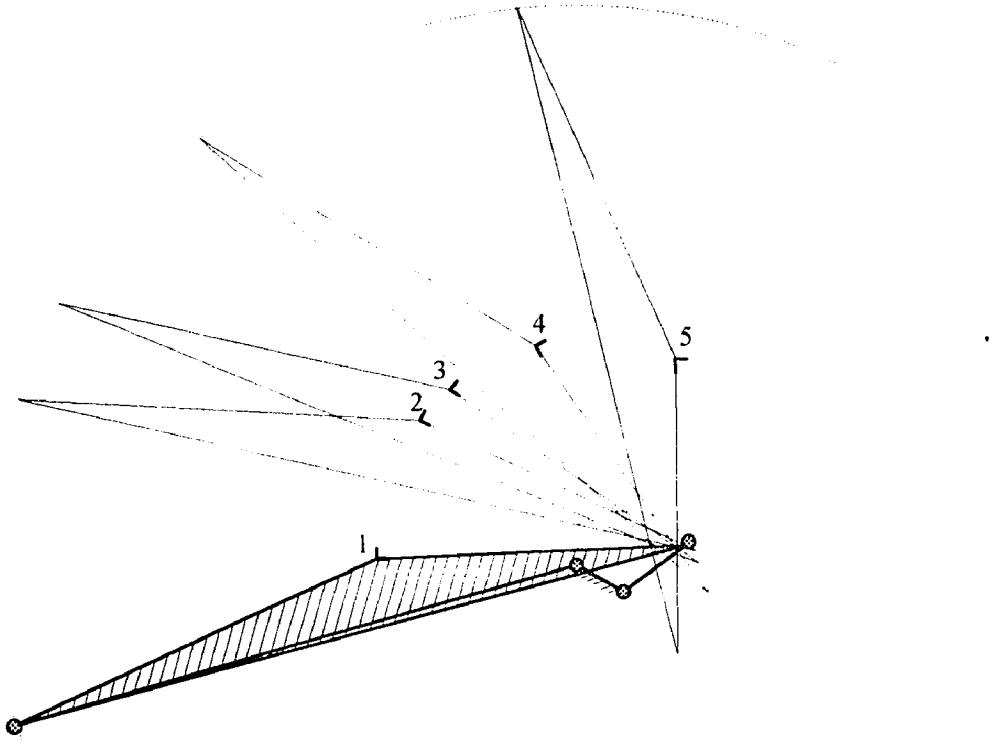
The convergence condition is satisfied, if on two successive iterations the residual sum of square estimates have relative difference less than or equal to EPS. The final convergence criterion is DELTA. The convergence condition is satisfied, if on two successive iterations the (Euclidean) norm of the approximate gradient is less than or equal to DELTA. In all of the examples in chapter 6, the EPS criterion was the effective one. The effect of different values of EPS is shown in Table 6-6 and 6-7. The appropriate values for EPS are between 10^{-5} and 10^{-8} in our tests. The total errors in the examples are between $9x10^{-3}$ and $1x10^{-4}$ which generally are smaller than the deflections of the links or the backlashes of the joints for application purposes. By decreasing the value of EPS, the calculations converge to a better solution, but the number of iterations increase appreciably, for example, the number of iterations in case 1 and case 2 of Table 6-6 are 24,450 and 5,447, while in Table 6-7 the number of iterations become 34,195 and 197,931, respectively. The program may fail to converge if the stopping criteria are too small.

7.2 Future Work

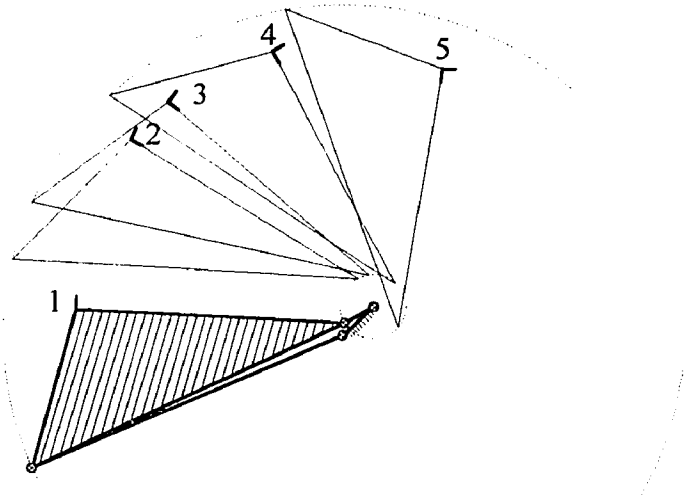
The proposed method is a general approach for motion synthesis of mechanisms. The use of coordinate transformations to form the constraint manifold equations provides a flexible way to formulate equations for motion synthesis problem and has potentials to solve for more complicated mechanisms. In the future work, we would like to apply this method for motion synthesis of some specialized three-dimensional mechanisms, for example, RSSR-SS mechanisms as discussed in chapter 3.

APPENDIX A

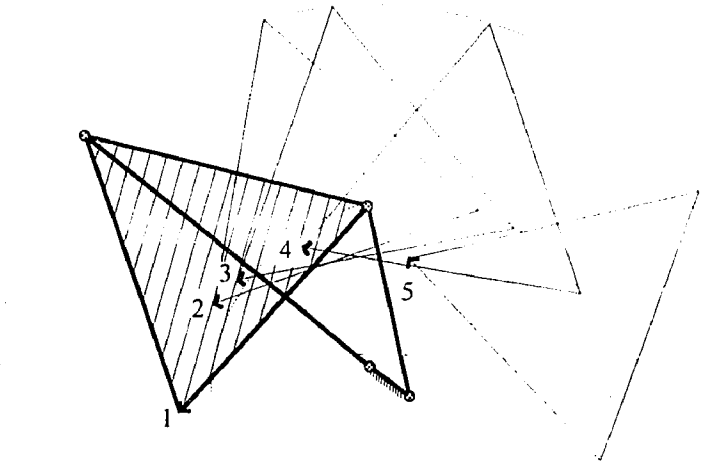
SOME OTHER SOLUTIONS FOR THE FIVE-POSITION EXAMPLE IN CHAPTER 6



Case 3



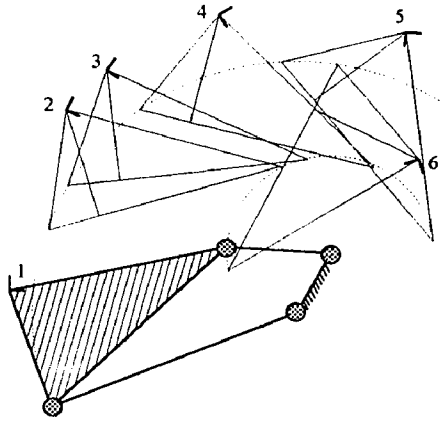
Case 4



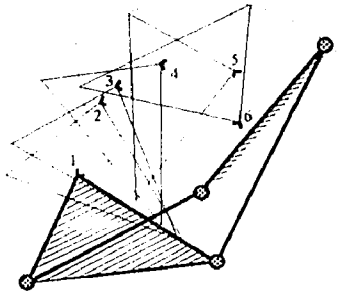
Case 5

APPENDIX B

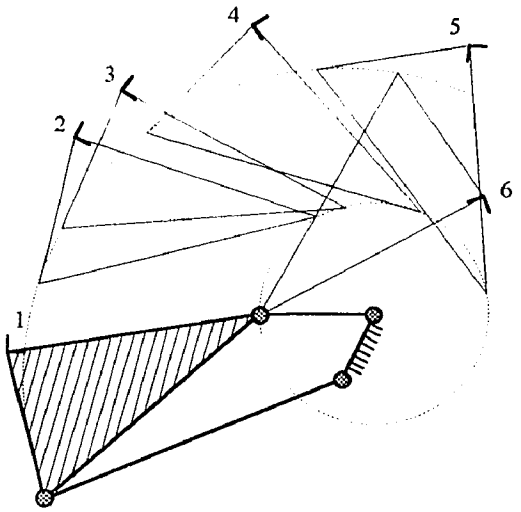
SOME OTHER SOLUTIONS FOR THE SIX-POSITION EXAMPLE IN CHAPTER 6



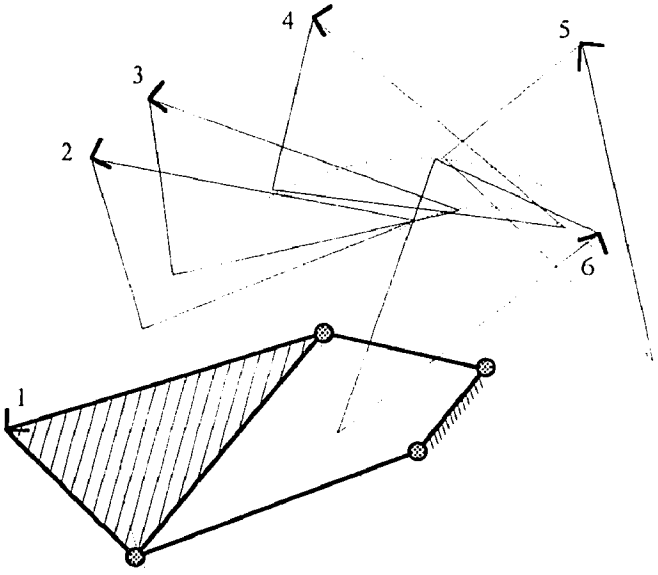
Case 5



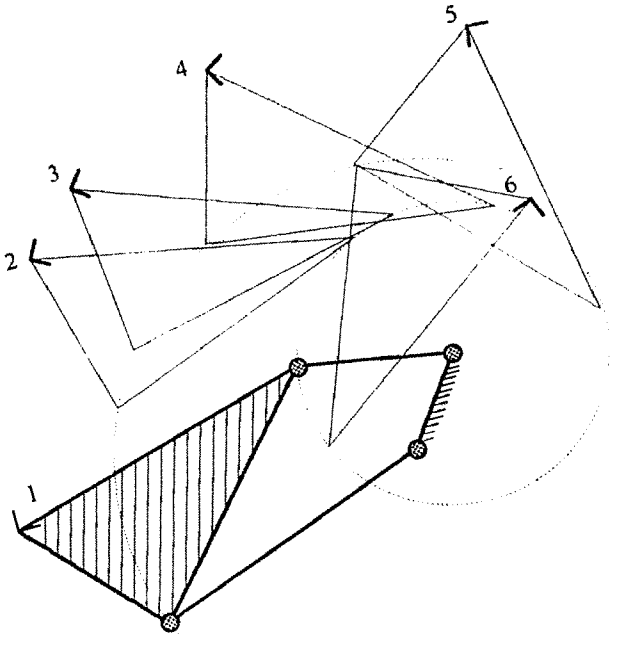
Case 6



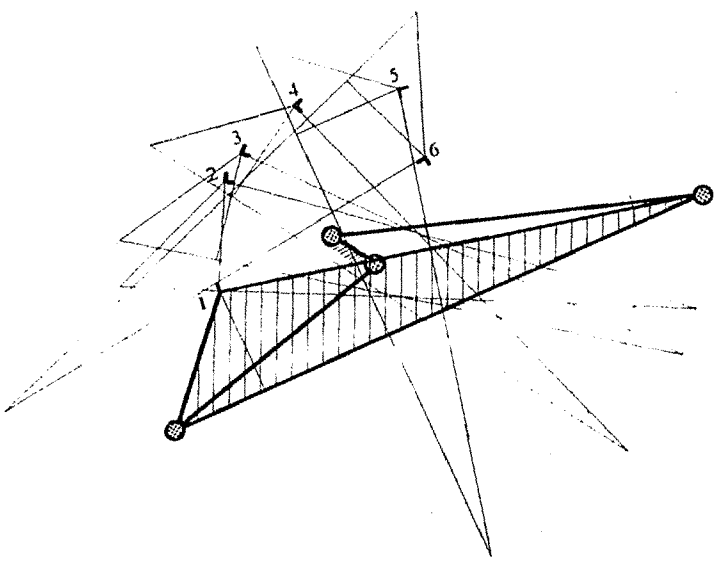
Case 7



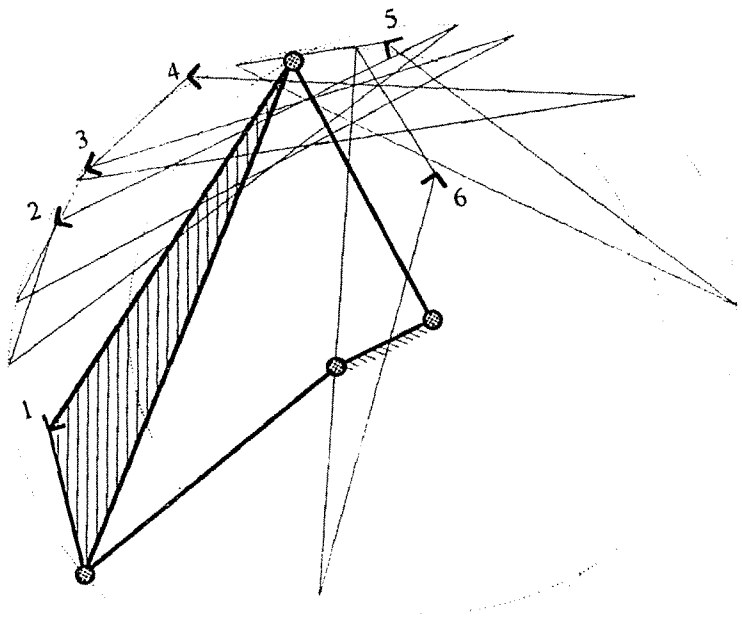
Case 8



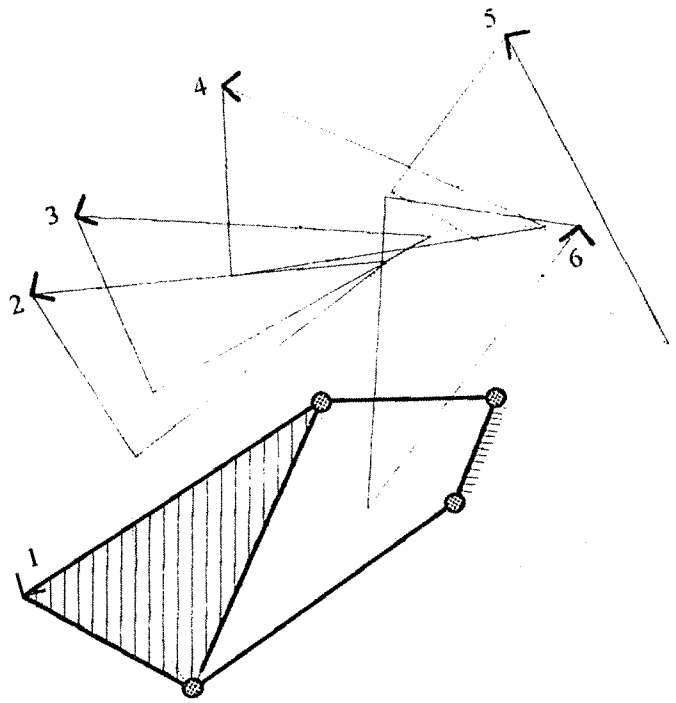
Case 9



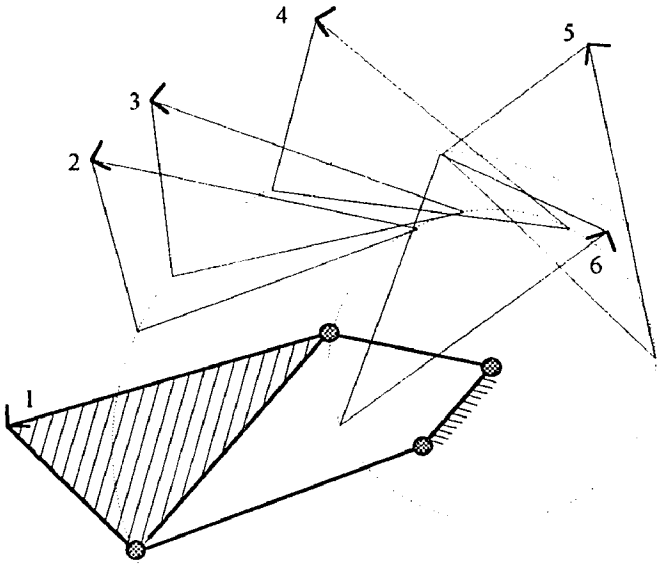
Case 10



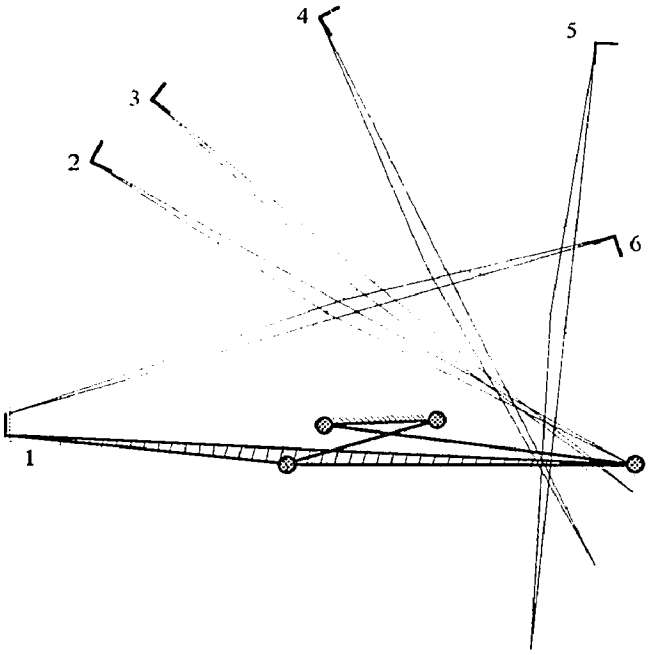
Case 11



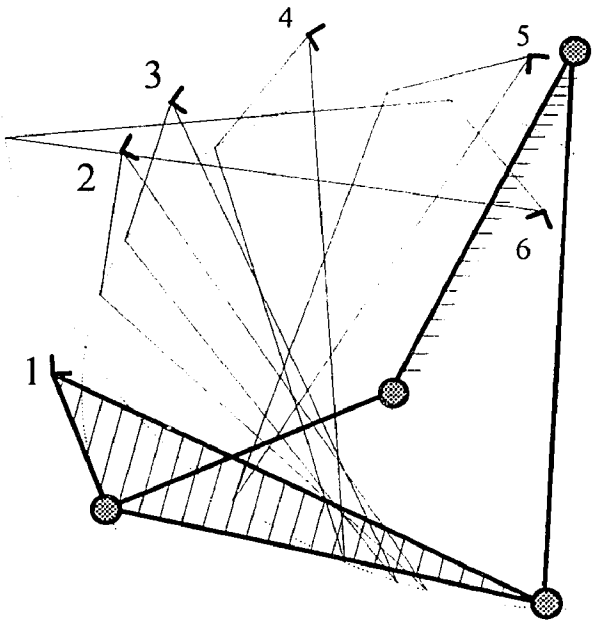
Case 12



Case 13



Case 14



Case 15

APPENDIX C

LIST OF THE FORTRAN PROGRAM FOR SYNTHESIS OF PLANAR MECHANISMS

```
C       A Fortran Program for
C       Motion Synthesis of a Planar Four-Bar Mechanism
C       using Constraint Manifolds Equation.
C
        implicit real*8 (a-h,o-z)
        real*8 x(4,10), r(10)
        common  x

        pi=4.0*atan(1.0)
C
        call precision ()
        call guess(r)
        call optimal(r)
        end

C*************************************************************
        subroutine precision()
C       Input desired positions and convert to quaternions
C*************************************************************
        implicit real*8 (a-h,o-z)
        real*8 phi(10), a(10), b(10),x(4,10)
        common x
C
        pi=4.0*atan(1.0)
        degtorad=pi/180.
C
        OPEN(UNIT=5,FILE='gplnr6.dat',STATUS='old')

C       Read in point#, orientations, and locations
        write (30,*) '    x1      x2      x3      x4'
        read (5,*)  m
        do 1 i=1,m
            read (5,*) k, phi(i), a(i), b(i)
            phi(i)=degtorad*phi(i)

C convert to quaternions
        x(1,i)= .5*a(i)*dcos(.5*phi(i))+.5*b(i)*dsin(.5*phi(i))
```

```
      x(2,i)=-.5*a(i)*dsin(.5*phi(i))+.5*b(i)*dcos(.5*phi(i))
      x(3,i)=dsin(0.5*phi(i))
      x(4,i)=dcos(0.5*phi(i))

      write (30,2) x(1,i),x(2,i),x(3,i),x(4,i)
1     continue
2     format (4F9.3)
      close(5)
      return
      end

C*************************************************
      subroutine guess(r)
C     Initial guess for the optimization routine
C*************************************************
      implicit real*8 (a-h,o-z)
      real*8  r(10)

      pi=4.0*atan(1.0)
      degtorad=pi/180.0

      OPEN(UNIT=5,FILE='guesse.dat',STATUS='old')

      read(5,*) x1, y1, a1, b1
      read(5,*) x2, y2, a2, b2
      read(5,*) h, eta
      close(5)

      eta=eta*degtorad

      r(1)=x1
      r(2)=y1
      r(3)=a1
      r(4)=b1

      r(5)=x2
      r(6)=y2
      r(7)=a2
      r(8)=b2

      r(9)=h
      r(10)=eta

      return
      end
```

```
C*******************************************************
      subroutine optimal(r)
C  Modified from the example of IMSL packages
C*******************************************************
      implicit real*8 (a-h,o-z)
      INTEGER   ldfjac, m, n
      PARAMETER  (ldfjac=6, m=6, n=10)
      real*8  FJAC(LDFJAC,N), FVEC(M), RPARAM(7),r(N), work(1000),
     & xjtj(500)
      EXTERNAL  func

      call zxssq(func, m, n, 5, 1.0D-6, 1.0D-6, 1000, 0, rparam,
     &  r, ssq, fvec, fjac, ldfjac, xjtj,
     & work, infer, ier)

C convert design variables back to angles

      pi=4.0*atan(1.0)
      r(10)=r(10)*180./pi

      WRITE (30,99999) r, FVEC,work(5),work(2), ssq, infer
99999  FORMAT ('  The solution is ', 10F9.4, //, '  The function ',
     & 'evaluated at the solution is ', /, 18X, 6D9.4, //,
     & '  The number of iterations is ', 10X, F9.2, /, '  The ',
     & 'number of function evaluations is ', F9.2, /,
     & 'The total error=',D9.4/,
     & 'The convergence parameters=',I2/)
      return
      end
C
C*******************************************************
      Subroutine func(r,m,n,f)
C*******************************************************
      implicit real*8 (a-h,o-z)
      integer m,n
      real*8 r(n),f(m)
      real*8 Q1(4,4),Q2(4,4),g1(4,4),G2(4,4),H1(4,4),H2(4,4)
      real*8 tmp1(4,4),c1(4,4),c1invt(4,4),c2(4,4),c2invt(4,4)
      real*8 Rzeta(4,4)
      real*8 v(3),ja(3,4),u(4),jat(4,3)
      real*8 jajat(3,3),temp(3),x(4,10)
      common x

C   r(1)=x1, r(2)=y1, r(3)=a1, r(4)=b1,
C   r(5)=x2, r(6)=y2, r(7)=a2, r(8)=b2,
```

```
C   r(9)=h, r(10)=eta

      pi=4.0*atan(1.0)

C Initialize the arrays

      do 1, i=1,4
         do 1, j=1,4

            Q1(i,j)=0.0
            Q2(i,j)=0.0
            G1(i,j)=0.0
            G2(i,j)=0.0
            H1(i,j)=0.0
            H2(i,j)=0.0
1           Rzeta(i,j)=0.0

      Q1(1,1) = 1.
      Q1(2,2) = 1.
      Q1(3,3) =-r(3)**2/4.
      Q1(4,4) = Q1(3,3)

      Q2(1,1) = 1.
      Q2(2,2) = 1.
      Q2(3,3) =-r(7)**2/4.
      Q2(4,4) = Q2(3,3)

      G1(1,1) = 1.
      G1(2,2) = 1.
      G1(3,3) = 1.
      G1(4,4) = 1.
      G1(1,3) = r(2)/2.
      G1(1,4) = r(1)/2.
      G1(2,3) =-r(1)/2.
      G1(2,4) = r(2)/2.

      G2(1,1) = 1.
      G2(2,2) = 1.
      G2(3,3) = 1.
      G2(4,4) = 1.
      G2(1,3) = r(6)/2.
      G2(1,4) = r(5)/2.
      G2(2,3) =-r(5)/2.
      G2(2,4) = r(6)/2.
```

```
      Rzeta(1,1)=dcos(0.5*r(10))
      Rzeta(2,2)=Rzeta(1,1)
      Rzeta(3,3)=Rzeta(1,1)
      Rzeta(4,4)=Rzeta(1,1)
      Rzeta(1,2)=dsin(0.5*r(10))
      Rzeta(2,1)=-Rzeta(1,2)
      Rzeta(3,4)= Rzeta(1,2)
      Rzeta(4,3)= Rzeta(2,1)

      H1(1,1) = 1.
      H1(2,2) = 1.
      H1(3,3) = 1.
      H1(4,4) = 1.
      H1(1,3) =-r(9)/2.
      H1(1,4) = r(4)/2.
      H1(2,3) = r(4)/2.
      H1(2,4) = r(9)/2.

      H2(1,1) = 1.
      H2(2,2) = 1.
      H2(3,3) = 1.
      H2(4,4) = 1.
      H2(1,3) =-r(9)/2.
      H2(1,4) =-r(8)/2.
      H2(2,3) =-r(8)/2.
      H2(2,4) = r(9)/2.

      call matmult(g1,rzeta,tmp1,4,4,4)
      call matmult(tmp1,h1,c1,4,4,4)
      call inv(c1,4)
      call matransp(c1,4,4,c1invt)
      call matmult(c1invt,Q1,tmp1,4,4,4)
      call matmult(tmp1,c1,Q1,4,4,4)

      call matmult(G2,Rzeta,tmp1,4,4,4)
      call matmult(tmp1,H2,c2,4,4,4)
      call inv(c2,4)
      call matransp(c2,4,4,c2invt)
      call matmult(c2invt,Q2,tmp1,4,4,4)
      call matmult(tmp1,c2,Q2,4,4,4)

      do 222, k=1,M

         v(1)=-(Q1(1,1)*x(1,k)**2+2.*Q1(1,2)*x(1,k)*x(2,k)
     &        +2.*Q1(1,3)*x(1,k)*x(3,k)+2.*Q1(1,4)*x(1,k)*x(4,k)
```

```
     &          +Q1(2,2)*x(2,k)**2+2.*Q1(2,3)*x(2,k)*x(3,k)
     &          +2.*Q1(2,4)*x(2,k)*x(4,k)+Q1(3,3)*x(3,k)**2
     &          +2.*Q1(3,4)*x(3,k)*x(4,k)+Q1(4,4)*x(4,k)**2)

          v(2)=-(Q2(1,1)*x(1,k)**2+2.*Q2(1,2)*x(1,k)*x(2,k)
     &          +2.*Q2(1,3)*x(1,k)*x(3,k)+2.*Q2(1,4)*x(1,k)*x(4,k)
     &          +Q2(2,2)*x(2,k)**2+2.*Q2(2,3)*x(2,k)*x(3,k)
     &          +2.*Q2(2,4)*x(2,k)*x(4,k)+Q2(3,3)*x(3,k)**2
     &          +2.*Q2(3,4)*x(3,k)*x(4,k)+Q2(4,4)*x(4,k)**2)

        v(3)=0.0

        ja(1,1)=0.0
        ja(1,2)=0.0
        ja(1,3)=0.0
        ja(1,4)=0.0
        ja(2,1)=0.0
        ja(2,2)=0.0
        ja(2,3)=0.0
        ja(2,4)=0.0

        do 3, i=1,4
           ja(1,1)=ja(1,1)+2.*Q1(1,i)*x(i,k)
           ja(1,2)=ja(1,2)+2.*Q1(2,i)*x(i,k)
           ja(1,3)=ja(1,3)+2.*Q1(3,i)*x(i,k)
           ja(1,4)=ja(1,4)+2.*Q1(4,i)*x(i,k)
           ja(2,1)=ja(2,1)+2.*Q2(1,i)*x(i,k)
           ja(2,2)=ja(2,2)+2.*Q2(2,i)*x(i,k)
           ja(2,3)=ja(2,3)+2.*Q2(3,i)*x(i,k)
3          ja(2,4)=ja(2,4)+2.*Q2(4,i)*x(i,k)

        ja(3,1)=0.0
        ja(3,2)=0.0
        ja(3,3)=2.*x(3,k)
        ja(3,4)=2.*x(4,k)

C       Calculate the inverse of J matrix
        call matransp(ja, 3, 4,jat)
        call matmult(ja,jat,jajat,3,4,3)
        call inv(jajat,3)

C The normal error vector u is the product of Jacobian inverse and v vector
        call matxvec(jajat, 3, 3, v, temp)
        call matxvec(jat, 4, 3, temp, u)
```

```
C The error function eq.24
      f(k) = vecxvec(u, u, 4)

222   continue
      err=0.0

C     Add penalty functions to r(3) and r(7)
      IF (r(3).LE.0.0) THEN
      err=err+1000000.0
      endif
      IF (r(7).LE.0.0) THEN
      err=err+1000000.0
      endif
      do 123, k=1,M
          f(k)=err+f(k)
 123      f(k)=sqrt(f(k))
C     write(6,*) 'Total err = ',err
      return
      end

C*************************************************************
      subroutine matmult(a, b, c, l, m, n)
C*************************************************************
      real*8  a(l,m), b(m,n), c(l,n)

      do 1, i=1,l
          do 1, j=1,n
              c(i,j)=0.0
      do 1, k=1,m
  1       c(i,j) = c(i,j)+a(i,k)*b(k,j)
      return
      end

C*************************************************************
      subroutine matransp(a,m,n,b)
C*************************************************************
      real*8  a(m,n), b(n,m)

      do 1, i=1,m
          do 1,j=1,n
  1           b(j,i)=a(i,j)
      return
      end
```

```
C*************************************************************
      SUBROUTINE inv(a, mm)
C*************************************************************
      integer mm, i, j, k, m,  nn
      real*8 a(mm,mm), b(10), c(10)
      real*8 d

      nn = mm - 1
      a(1,1) = 1.0 / a(1,1)
      do 400 m = 1, nn
         k = m + 1
         do 410 i = 1, m
            b(i) = 0.0
            do 410 j = 1, m
410            b(i) = b(i) + a(i,j) * a(j,k)
      d = 0.0
      do 420 i = 1, m
420      d = d + a(k,i) * b(i)
      d = -1.0 * d + a(k,k)
      a(k,k) = 1.0 / d
      do 430 i = 1, m
430      a(i,k) = -1.0 * b(i) * a(k,k)
      do 440 j = 1, m
         c(j) = 0.0
         do 440 i = 1, m
440            c(j) = c(j) + a(k,i) * a(i,j)
      do 450 j = 1, m
450      a(k,j) = -1.0 * c(j) * a(k,k)
      do 460 i = 1, m
         do 460 j = 1, m
460            a(i,j) = a(i,j) - b(i) * a(k,j)
400   continue
      return
      end
C*************************************************************
      subroutine matxvec(a, m, n, b, c)
C*************************************************************
      real*8 a(m,n), b(n), c(n)

      do 1, i = 1,m
         c(i) = 0.0
         do 1, j=1,n
1           c(i) = c(i) + a(i, j)*b(j)
      return
      end
```

```
C*************************************************************
      subroutine vecxmat(a, m, n, b, c)
C*************************************************************
      real*8 a(m, n), b(m), c(m)

      do 1, j=1,n
         c(j) = 0.0
      do 1, i=1,m
    1    c(j) = c(j) + b(j) * a(j, i)
      return
      end

C*************************************************************
      function vecxvec(a, b, n)
C*************************************************************
      implicit real*8 (a-h,o-z)
       real*8 a(n), b(n)
      temp = 0.
      do 1, i=1,n
    1    temp=temp+a(i)*b(i)
      vecxvec = temp
      return
      end
```

APPENDIX D

LIST OF THE COST FUNCTION SUBROUTINE FOR THE ASA ALGORITHM

The Cost function subroutine written in *C* for the ASA algorithm used to synthesize spherical four-bar mechanisms are listed in the following:

```
/***********************************************************************
* double cost_function
*       This is the users cost function to optimize
*       (find the minimum).
*       cost_flag is set to true if the parameter set
*       does not violates any constraints
*       parameter_lower_bound and parameter_upper_bound may be
*       adaptively changed during the search.
***********************************************************************/
#if HAVE_ANSI
double cost_function(double *x,
                     double *parameter_lower_bound,
                     double *parameter_upper_bound,
                     double *cost_tangents,
                     double *cost_curvature,
                     ALLOC_INT * parameter_dimension,
                     int *parameter_int_real,
                     int *cost_flag,
                     int *exit_code,
                     USER_DEFINES * USER_OPTIONS)
#else
double cost_function(x,
                     parameter_lower_bound,
                     parameter_upper_bound,
                     cost_tangents,
                     cost_curvature,
                     parameter_dimension,
                     parameter_int_real,
                     cost_flag,
                     exit_code,
                     USER_OPTIONS)
double *x;
double *parameter_lower_bound;
double *parameter_upper_bound;
```

```
double *cost_tangents;
double *cost_curvature;
ALLOC_INT *parameter_dimension;
int *parameter_int_real;
int *cost_flag;
int *exit_code;
USER_DEFINES *USER_OPTIONS;
#endif
{
#if ASA_TEST          /* ASA test problem */
#if SELF_OPTIMIZE
#else
    static LONG_INT funevals = 0;
#endif
    double s[3][10],xd[4][10];
    double phi[10];
    double Q1[4][4],Q2[4][4],Rzpsi1[4][4],Rzpsi2[4][4];
    double Rxal1[4][4],Rxal2[4][4],Rxgam1[4][4],Rxgam2[4][4];
    double Ryseg[4][4],Rzeta[4][4],tmp1[4][4],tmp2[4][4],tmp3[4][4];
    double c1[4][4],c2[4][4],c1t[4][4],c2t[4][4];
    double v[3],ja[3][4],u[4],jat[4][3],jajat[3][3],jajatinv[3][3];
    double temp[3],f[10];
    double pow(), sum;
    FILE   *outfp, *infp;
    int    i,j,k,num_pos;

    if ((infp=fopen("in","r"))==NULL)
    {
      fprintf(stderr,"Input file not found");
      exit(-1);
    }
    outfp=fopen("out","w");
/*  fprintf(outfp, " Enter the number of position \n");*/
    fscanf(infp,"%d",&num_pos);
/*  fprintf(outfp, " Enter pos#, directional cosine, rotation angle\n");  */
    for(i=0; i<num_pos; i++)
      {
        fscanf(infp,"%d %lf %lf %lf %lf",&k,&s[0][i],&s[1][i],&s[2][i],&phi[i]);
        phi[i]=Degtorad*phi[i];
        for( j=0; j<3; j++) {
          xd[j][i]= s[j][i]*sin(0.5*phi[i]);}
          xd[3][i]=cos(0.5*phi[i]);
          fprintf(outfp,"%f  %f  %f %f\n",xd[0][i],xd[1][i],xd[2][i],xd[3][i]);
      }
```

```
    fclose(infp);
    fclose(outfp);

/* Initialize the matrices */
    for(i=0; i<4; i++)
      {
      for(j=0; j<4; j++)
        {
          Q1[i][j]=0.0;
          Q2[i][j]=0.0;
          Rzpsi1[i][j]=0.0;
          Rzpsi2[i][j]=0.0;
          Rxal1[i][j]=0.0;
          Rxal2[i][j]=0.0;
          Rxgam1[i][j]=0.0;
          Rxgam2[i][j]=0.0;
          Ryseg[i][j]=0.0;
          Rzeta[i][j]=0.0;}}
/*
        x[0]=psi1, x[1]=alpha1, x[2]=beta1, x[3]=gam1,
        x[4]=psi2, x[5]=alpha2, x[6]=beta2, x[7]=gam2,
        x[8]=seg,  x[9]=eta.
*/

        Q1[0][0]=cos(0.5*x[2])*cos(0.5*x[2]);
        Q1[1][1]=Q1[0][0];
        Q1[2][2]=-sin(0.5*x[2])*sin(0.5*x[2]);
        Q1[3][3]= Q1[2][2];

        Q2[0][0]=cos(0.5*x[6])*cos(0.5*x[6]);
        Q2[1][1]=Q2[0][0];
        Q2[2][2]=-sin(0.5*x[6])*sin(0.5*x[6]);
        Q2[3][3]= Q2[2][2];

        Rzpsi1[0][0]=cos(0.5*x[0]);
        Rzpsi1[1][1]=Rzpsi1[0][0];
        Rzpsi1[2][2]=Rzpsi1[0][0];
        Rzpsi1[3][3]=Rzpsi1[0][0];
        Rzpsi1[1][0]=sin(0.5*x[0]);
        Rzpsi1[0][1]=-Rzpsi1[1][0];
        Rzpsi1[2][3]= Rzpsi1[1][0];
        Rzpsi1[3][2]= Rzpsi1[0][1];

        Rzpsi2[0][0]=cos(0.5*x[4]);
        Rzpsi2[1][1]=Rzpsi2[0][0];
```

```
Rzpsi2[2][2]=Rzpsi2[0][0];
Rzpsi2[3][3]=Rzpsi2[0][0];
Rzpsi2[1][0]=sin(0.5*x[4]);
Rzpsi2[0][1]=-Rzpsi2[1][0];
Rzpsi2[2][3]= Rzpsi2[1][0];
Rzpsi2[3][2]= Rzpsi2[0][1];

Rxal1[0][0]=cos(0.5*x[1]);
Rxal1[1][1]=Rxal1[0][0];
Rxal1[2][2]=Rxal1[0][0];
Rxal1[3][3]=Rxal1[0][0];
Rxal1[0][3]=sin(0.5*x[1]);
Rxal1[1][2]=-Rxal1[0][3];
Rxal1[2][1]= Rxal1[0][3];
Rxal1[3][0]= Rxal1[1][2];

Rxal2[0][0]=cos(0.5*x[5]);
Rxal2[1][1]=Rxal2[0][0];
Rxal2[2][2]=Rxal2[0][0];
Rxal2[3][3]=Rxal2[0][0];
Rxal2[0][3]=sin(0.5*x[5]);
Rxal2[1][2]=-Rxal2[0][3];
Rxal2[2][1]= Rxal2[0][3];
Rxal2[3][0]= Rxal2[1][2];

Rzeta[0][0]=cos(0.5*x[9]);
Rzeta[1][1]= Rzeta[0][0];
Rzeta[2][2]= Rzeta[0][0];
Rzeta[3][3]= Rzeta[0][0];
Rzeta[0][1]=sin(0.5*x[9]);
Rzeta[1][0]=-Rzeta[0][1];
Rzeta[2][3]= Rzeta[0][1];
Rzeta[3][2]= Rzeta[1][0];

Ryseg[0][0]=cos(0.5*x[8]);
Ryseg[1][1]=Ryseg[0][0];
Ryseg[2][2]=Ryseg[0][0];
Ryseg[3][3]=Ryseg[0][0];
Ryseg[1][3]= sin(0.5*x[8]);
Ryseg[2][0]= Ryseg[1][3];
Ryseg[0][2]=-Ryseg[1][3];
Ryseg[3][1]= Ryseg[0][2];

Rxgam1[0][0]=cos(0.5*x[3]);
Rxgam1[1][1]= Rxgam1[0][0];
```

```
Rxgam1[2][2]= Rxgam1[0][0];
Rxgam1[3][3]= Rxgam1[0][0];
Rxgam1[0][3]=sin(0.5*x[3]);
Rxgam1[1][2]= Rxgam1[0][3];
Rxgam1[2][1]=-Rxgam1[0][3];
Rxgam1[3][0]= Rxgam1[2][1];

Rxgam2[0][0]=cos(0.5*x[7]);
Rxgam2[1][1]=Rxgam2[0][0];
Rxgam2[2][2]=Rxgam2[0][0];
Rxgam2[3][3]=Rxgam2[0][0];
Rxgam2[0][3]=-sin(0.5*x[7]);
Rxgam2[1][2]= Rxgam2[0][3];
Rxgam2[2][1]=-Rxgam2[0][3];
Rxgam2[3][0]= Rxgam2[2][1];

matmult444(Rzpsi1,Rxal1,tmp1,4,4,4);
matmult444(tmp1,Rzeta,tmp2,4,4,4);
matmult444(tmp2,Ryseg,tmp3,4,4,4);
matmult444(tmp3,Rxgam1,c1,4,4,4);
matransp44(c1,4,4,c1t);
matmult444(c1,Q1,tmp1,4,4,4);
matmult444(tmp1,c1t,Q1,4,4,4);

matmult444(Rzpsi2,Rxal2,tmp1,4,4,4);
matmult444(tmp1,Rzeta,tmp2,4,4,4);
matmult444(tmp2,Ryseg,tmp3,4,4,4);
matmult444(tmp3,Rxgam2,c2,4,4,4);
matransp44(c2,4,4,c2t);
matmult444(c2,Q2,tmp1,4,4,4);
matmult444(tmp1,c2t,Q2,4,4,4);

for(k=0;k<num_pos; k++)
  { v[0]=-(Q1[0][0]*pow(xd[0][k],2.)+2.*Q1[0][1]*xd[0][k]*xd[1][k]
    +2.*Q1[0][2]*xd[0][k]*xd[2][k]+2.*Q1[0][3]*xd[0][k]*xd[3][k]
    +Q1[1][1]*pow(xd[1][k],2.)+2.*Q1[1][2]*xd[1][k]*xd[2][k]
    +2.*Q1[1][3]*xd[1][k]*xd[3][k]+Q1[2][2]*pow(xd[2][k],2.)
    +2.*Q1[2][3]*xd[2][k]*xd[3][k]+Q1[3][3]*pow(xd[3][k],2.));

    v[1]=-(Q2[0][0]*pow(xd[0][k],2.)+2.*Q2[0][1]*xd[0][k]*xd[1][k]
    +2.*Q2[0][2]*xd[0][k]*xd[2][k]+2.*Q2[0][3]*xd[0][k]*xd[3][k]
    +Q2[1][1]*pow(xd[1][k],2.)+2.*Q2[1][2]*xd[1][k]*xd[2][k]
    +2.*Q2[1][3]*xd[1][k]*xd[3][k]+Q2[2][2]*pow(xd[2][k],2.)
    +2.*Q2[2][3]*xd[2][k]*xd[3][k]+Q2[3][3]*pow(xd[3][k],2.));
v[2] = 0.0;
```

```
        ja[0][0]=0.0;
        ja[0][1]=0.0;
        ja[0][2]=0.0;
        ja[0][3]=0.0;
        ja[1][0]=0.0;
        ja[1][1]=0.0;
        ja[1][2]=0.0;
        ja[1][3]=0.0;

        for(i=0;i<4;i++)
          { ja[0][0]=ja[0][0]+2.*Q1[0][i]*xd[i][k];
            ja[0][1]=ja[0][1]+2.*Q1[1][i]*xd[i][k];
            ja[0][2]=ja[0][2]+2.*Q1[2][i]*xd[i][k];
            ja[0][3]=ja[0][3]+2.*Q1[3][i]*xd[i][k];
            ja[1][0]=ja[1][0]+2.*Q2[0][i]*xd[i][k];
            ja[1][1]=ja[1][1]+2.*Q2[1][i]*xd[i][k];
            ja[1][2]=ja[1][2]+2.*Q2[2][i]*xd[i][k];
            ja[1][3]=ja[1][3]+2.*Q2[3][i]*xd[i][k];}

        ja[2][0]=2.*xd[0][k];
        ja[2][1]=2.*xd[1][k];
        ja[2][2]=2.*xd[2][k];
        ja[2][3]=2.*xd[3][k];

        matransp34(ja, 3, 4,jat);
        matmult343(ja,jat,jajat,3,4,3);
        inv3(jajat,3);
        matxvec33(jajat, 3, 3, v, temp);
        matxvec43(jat, 4, 3, temp, u);
        f[k]=0.0;
        for(j=0;j<4;j++){
            f[k]=f[k]+u[j]*u[j];
        }
/*   f(k) = vecxvec(u, u, 4); */
        }
        sum = 0.0;
/* Add penalty function to beta1 and beta2 */
        if (x[3] <=0.0)
        sum+=1000000.;
        if (x[6] <=0.0)
        sum+=1000000.;

        for (k=0;k< num_pos;k++){
            sum+=f[k];
```

```
        }
/*
        fprintf(stderr," sum=%f\n",sum);
*/
        funevals += 1;
        *cost_flag = TRUE;

#if SELF_OPTIMIZE
#else
#if TIME_CALC
        /* print the time every PRINT_FREQUENCY evaluations */
        if ((PRINT_FREQUENCY > 0) && ((funevals % PRINT_FREQUENCY) == 0))
{
        fprintf(ptr_out, "funevals = %ld  ", funevals);
        print_time("");
        }
#endif
#endif
        return (sum);
#endif                          /* ASA_TEST */
} /* cost function */

matmult444(a, b, c, l, m, n)
int     l, m, n;
double a[4][4], b[4][4], c[4][4];
{
 int i,j,k;

  for(i=0;i<l;i++){
    for(j=0;j<n;j++){
        c[i][j]=0.0;
        for(k=0;k<m;k++){
          c[i][j]=c[i][j]+a[i][k]*b[k][j];
        }
    }
  }
}

matmult343(a, b, c, l, m, n)
int   l, m, n;
double a[3][4], b[4][3], c[3][3];
{
 int i,j,k;

  for(i=0;i<l;i++){
```

```
    for(j=0;j<n;j++){
        c[i][j]=0.0;
        for(k=0;k<m;k++){
          c[i][j]=c[i][j]+a[i][k]*b[k][j];
        }
    }
  }
}

matransp44(a, m, n, b)
int m, n;
double a[4][4], b[4][4];
{
  int  i,j;

  for(i=0;i<m;i++){
    for(j=0;j<n;j++){
        b[j][i]=a[i][j];
    }
  }
}

matransp34(a, m, n, b)
int m, n;
double a[3][4], b[4][3];
{
  int  i,j;

  for(i=0;i<m;i++){
    for(j=0;j<n;j++){
        b[j][i]=a[i][j];
    }
  }
}

inv4( a, mm)
int mm;
double a[4][4];
{
    int i,j,k,m,nn;
    double b[10],c[10];
    double d;

    if (a[0][0] ==0.0) {
    printf(" a[0][0]= 0.0  Singular Matrix \n");
```

```
    for(i=0;i<mm;i++) {
      for(j=0;j<mm;j++){
        a[i][j]=0.0;
      }
    }
    exit(0);
    }
    else
    { nn=mm-1;
    a[0][0]=1.0/a[0][0];
    for (m=0;m<nn;m++){
        k = m+1;
        for(i=0;i<=m;i++){
          b[i]=0.0;
          for (j=0;j<=m;j++){
            b[i]=b[i]+a[i][j]*a[j][k];
          }
        }
    d = 0.0;
    for(i=0;i<=m;i++){
      d = d +a[k][i]*b[i];
    }
    d = -1.0*d +a[k][k];
    a[k][k]=1.0/d;
    for(i=0;i<=m;i++){
      a[i][k]=-1.0*b[i]*a[k][k];
    }
    for(j=0;j<=m;j++){
      c[j]=0.0;
      for(i=0;i<=m;i++){
        c[j]=c[j]+a[k][i]*a[i][j];
      }
    }
    for(j=0;j<=m;j++){
        a[k][j]=-1.0*c[j]*a[k][k];}

    for (i=0;i<=m;i++){
        for(j=0;j<=m;j++){
        a[i][j]=a[i][j]-b[i]*a[k][j];
     }
    }
   }
  }
}
```

```
matxvec33(a, m, n, b, c)
int  m,n;
double a[3][3], b[3], c[3];
{
  int i,j;

  for(i=0;i<m;i++){
    c[i]=0.0;
    for(j=0;j<n;j++){
      c[i]=c[i]+a[i][j]*b[j];
      }
    }
}

matxvec43(a, m, n, b, c)
int  m,n;
double a[4][3], b[3], c[4];
{
  int i,j;

  for(i=0;i<m;i++){
    c[i]=0.0;
    for(j=0;j<n;j++){
      c[i]=c[i]+a[i][j]*b[j];
      }
    }
}
```

14. Fletcher, R., 1987, *Practical Methods of Optimization*, 2nd edition, John Wiley and Sons, New York, 436pp.

15. Freudenstein, F. and G. N. Sandor, "On the Burmester Points of a Plane," *Journal of Applied Mechanics*, Transaction ASME Series E, Vol. 28, No. 1, March 1961 pp. 41-49.

16. Garner, L. E., 1981, *An Outline of Projective Geometry*, North Holland Publisher, Amsterdam, 220pp.

17. Ge, Q. J. and J. M. McCarthy, 1988a, "The Image Curve of the Planet in a Spherical Epicyclic Gear Train," ASME *Journal of Mechanisms, Transmissions, and Automation in Design*, V.110, pp.281-286.

18. Ge, Q. J. and J. M. McCarthy, 1988b, "Classification of the Image Curves of Spherical Four-Bar Linkages," ASME *Journal of Mechanisms, transmissions, and automation in design*, Vol. 110, No. 3, September.

19. Ge, Q. J. and J. M. McCarthy, 1989, "The Constraint Manifold of the Coupler of an SSSS Spatial Four Bar Linkage," *First National Applied Mechanisms and Robotics Conference*, November 5-8, 1989, Cincinnati, OH, AMR-3B-5.

20. Ge, Q. J. and B. Ravani, 1991, "Computer Aided Geometric Design of Motion Interpolants," DE-Vol. 32-2, ASME *Advances in Design Automation*, Vol. 2.

21. Hansen, M. R. 1992 "A General Procedure for Dimensional Synthesis for Mechanisms," ASME *Mechanical Design and Synthesis*, DE-Vol. 46, pp. 67-71.

22. Haug, E. J., 1992, *Intermediate Dynamics*, Prentice Hall, Englewood Cliffs, New Jersey. 420pp.

23. Jenuwine, J. G. and A. Mihda, "Synthesis of Single-input and Multiple-output Port Mechanisms with Springs for Specified Energy Absorption," ASME *Mechanical Design and Synthesis*, DE-Vol. 46, pp. 73-79.

24. Ingber, L., 1989, "Very Fast Simulated Re-annealing," *Mathematical and Computer Modelling*, vol. 12, no. 8, pp. 967-973.

25. Ingber, L. and B. Rosen, 1992, "Genetic Algorithm and Very Fast Simulated Reannealing: A Comparison," *Mathematical and Computer Modelling*, vol. 16, no. 11, pp.87-100.

26. Ingber, L., 1993, "*Adaptive Simulated Annealing* (ASA)," available via anonymous ftp from [*ftp.caltech.edu*: /pub/ingber/asa.Z], Lester Ingber Research, McLean, Virginia.

27. Leu, M. C., Wong H., and Ji, Z., 1993 "Planning of Component Placement/Insertion Sequence and Feeder Setup in PCB Assembly Using Genetic Algorithm" ASME, *Journal of Electronic Packaging*, Vol. 115, pp.424-432.

28. *Lincages-4 Manual*, 1988, University of Minnesota and Minnesata Technology Transfer (MINTT), Inc. 59pp.

29. McCarthy, J. M., 1983, "Planar and Spatial Rigid Body Motion as Special Cases of Spherical and 3-Spherical Motion," ASME *Journal of Mechanisms, Transmissions, and Automation in Design*, Vol. 105, No 3, pp. 569-575.

30. McCarthy, J. M., 1987, "The Differential Geometry of Curves in an Image Space of Spherical Kinematics," *Mechanism & Machine Theory*, Vol. 22, No. 3, pp. 205-211.

31. McCarthy, J. M., 1988, "The Image Curve of the Coupler of a Special Spherical Four Bar Linkage," ASME *Journal of Mechanisms, Transmissions, and Automation in Design*, V.110, pp. 276-280.

32. McCarthy, J. M., 1990, *Introduction to Theoretical Kinematics*, MIT Press, Cambridge, Massachusetts, 130pp.

33. Mirth, J. A., 1992, "A Complex Number Approach for Absolute Precision Position Synthesis for Three Precision Positions," ASME *Mechanical Design and Synthesis* DE-Vol. 46, pp. 43-48.

34. Nikravesh, P. E., 1988, *Computer-Aided Analysis of Mechanical Systems*. Prentice Hall International, 370pp.

35. Larochelle, P., J. Dooley, A. Murray, and J. M. McCarthy, 1993 "*Sphinx*-Software for Synthesizing Spherical 4R Mechanisms," *Proceedings of the 1993 NSF Design and Manufacturing Systems Conference*, Volume1, pp. 607-611.

36. Porteous, I. R., 1981 *Topological Geometry*, 2nd edition, Cambridge University Press, Cambridge, Massachusetts, 486pp.

37. Premkumar, P., S. R. Dhall, and S. N. Kramer, September, 1988, "Selective Precision Synthesis of the Spatial Slider Crank Mechanism for Path and Function Generation," ASME *Journal of Mechanism, Transmissions, and Automation in Design*, vol. 110, pp. 295-302.

38. Rao, S. S., 1984, *Optimization Theory and Application*, 2nd Edition, Wiley Eastern Limited, New Delhi, India, 747pp.

39. Ravani, B., 1982, *Kinematic Mappings as Applied to Motion Approximation and Mechanism Synthesis*, PhD thesis, Standford University, California.

40. Ravani, B. and B. Roth, 1983, "Motion Synthesis Using Kinematic Mappings," ASME *Journal of Mechanism, Transmissions, and Automation in Design*, Vol. 105, pp. 460-467, September.

41. Ravani, B. and B. Roth, 1984, "Mapping of Spatial Kinematics," ASME *Journal of Mechanisms, Transmissions, and Automation in Design*, Vol. 106, pp. 341-347.

42. Sandor, G. N., L Xu, and S. P. Yang, 1986, "Computer-Aided Synthesis of Two-closed-loop RSSR-SS Spatial Motion Generator with Branching and Sequence Constraints," *Mechanism & Machine Theory*, Vol. 21, No. 4, pp. 345-350.

43. Sander, G. N., T. C. Weng, and Y. Xu, 1988 "The Synthesis of Spatial Motion Generators with Prismatic, Revolute and Cylindric Pairs Without Branching Defect," *Mechanism & Machine Theory*, Vol. 23, No. 4, pp. 269-274.

44. Sander, G. N. and A. G. Erdman, 1988, *Advanced Mechanism Design: Analysis and Synthesis*, Vol. 2, 688.pp.

45. Schraudolph, N. N and J. J. Grefenstette, 1992, "*A User's Guide to GAucsd 1.4*," available via anonymous ftp from [*cs.ucsd.edu*(132.239.51.3) /pub/GAucsd], Report, University of California at San Diago, La Jolla, CA, 22 pp.

46. Subbian, T. and D. R. Flugrad, 1992 "Synthesis of an RSSR-SS Mechanism using a Continuation Method," ASME *Mechanical Design and Synthesis Conference*, DE-Vol. 46, pp. 579-586.

47. Suh, C. H. and C. W. Radcliffe, 1978, *Kinematics and Mechanisms Design*, John Wiley and Sons, New York, 434pp.

48. Sutherland, G. H., 1977, "Mixed Exact-Approximate Planar Mechanism Position Synthesis," ASME *Journal of Engineering for Industry*, pp. 434-439.

49. Wang, S. J. 1993, "*Kinematic Synthesis of Adjustable Crank Mechanisms for Three Phase Motion Generation*," PhD Dissertation, NJIT, Newark, New Jersey.

50. Wehage, R. A., 1984, "Quaternion and Euler Parameters - A Brief Exposition," *Computer Aided Analysis and Optimization of Mechanical System Dynamics*, edition E, J. Haug, Heidelburg: Spring-Verlag, pp. 147-180.

51. Wilhelm, A. J., 1989, *Kinematic Synthesis of Adjustable Linkages for Motion Generation*, PhD Dissertation, The Wichita State University, Wichita.

52. Yang A. T. and F. Freudenstein, 1964, "Application of Dual-Number Quaternion Algebra to the Analysis of Spatial Mechanisms," ASME *Journal of Applied Mechanics*, Vol. 86, No. 2, pp. 300-308.

53. Youm, Y. and T. C. Huang, 1990, "Exact Displacement Analysis of XCCC Spatial Mechanisms by the Directional Cosine Matrix Method," *Mechanism & Machine Theory*, Vol. 25, No. 1, pp. 85-96.